AFRL-RI-RS-TR-2011-057

MODELS OF QUALITY OF SERVICE AND QUALITY OF INFORMATION ASSURANCE TOWARDS THEIR DYNAMIC ADAPTATION

ARIZONA STATE UNIVERSITY

MARCH 2011

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2011-057 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/ /s/

PATRICK HURLEY
Work Unit Manager

WARREN H. DEBANY JR., Technical Advisor
Information Grid Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
March 2011	Final Technical Report	April 2008 – October 2010

4. TITLE AND SUBTITLE

MODELS OF QUALITY OF SERVICE AND QUALITY OF INFORMATION ASSURANCE TOWARDS THEIR DYNAMIC ADAPTATION

5a. CONTRACT NUMBER
FA8750-08-2-0155

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Nong Ye

5d. PROJECT NUMBER
459G

5e. TASK NUMBER
PH

5f. WORK UNIT NUMBER
01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Arizona State University
1711 S. Rural Road, ADM B160
Tempe AZ 85287-0002

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RIGA
525 Brooks Road
Rome NY 13441

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2011-057

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This report documents the details of the technical work accomplished by Arizona State University on defining Quality of Service (QoS) and Quality of Information Assurance (QoIA) models/metrics to enable intelligent QoS and QoIA trade-offs. ASU's task was to: (1) Establish QoS models/metrics and (2) Define and establish QoIA models/metrics. This report describes all of the research accomplished, information gained, techniques and procedures used in the research process applied during the performance of this project. It includes pertinent observations, discussion of problems encountered, positive and negative results, test case scenario design, and the results of the testing.

15. SUBJECT TERMS

Quality of Service, Quality of Information Assurance, computer security, network security,

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			PATRICK HURLEY
U	U	U	UU	70	19b. TELEPHONE NUMBER (*Include area code*) N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.

Abstract

Quality of Service (QoS) and Quality of Information Assurance (QoIA) for cyber security on computer and network systems have become increasingly important as many conventional service transactions are moved online. QoS of computer and network services can be measured by different aspects of service performance such as throughput, delay, and so on. On a computer and network system, competing service requests of users and associated service activities change the state of limited system resources which in turn affects the achieved QoS. Modeling dynamic relations of service activities, system state and QoS is required to determine if users' service requests and requirements of QoS can be satisfied by the system with limited resources and how the system and service configuration can be adapted to meet QoS requirements. QoIA is usually provided through security mechanisms such as encryption, intrusion detection, firewalls, etc.

An empirical approach is taken in this project to uncover these cause-effects dynamics of service activities, system state and QoS to built activity-state-quality (ASQ) models. These ASQ models capture activity-state-quality dynamics at a more comprehensive, system scale including multiple resources, their interactions, their state changes with service activities, and their effects on QoS. The empirical approach is applied to obtain ASQ models for a voice communication service. The results uncover four major types of dynamic relations among service activity parameters, system resource state, and the network throughput, an important measure of QoS for the voice communication service. The insights gained into the system dynamics together with the ASQ models can then be used to adapt the system and service configuration to meet QoS requirements.

The empirical approach is also used to uncover dynamic effects and tradeoffs of service, security and attack activities for two service scenarios: 1) a voice communication service with data encryption for information assurance and security, and 2) a motion detection service with intrusion detection for information assurance and security. For each of the service scenarios, the uncovered system dynamics are grouped into categories. The characteristics of the relations in

each category, tradeoff effects and their implications for QoS and QoIA adaptation and resource load balancing are reported. Additionally, system dynamics variables showing unique attack effects for both service scenarios were discovered. These variables can be used by the system to detect attacks. Some of these variables present unique characteristics to specific types of attacks. The discovered information enhances our understanding of the system dynamics and the interactions between services, security mechanisms and attacks.

In addition to the specific findings of ASQ dynamics and models for specific services, security mechanisms and attacks and implications of these findings for QoS and QoIA adaptations, a general approach to QoS and QoIA adaptations is found to raise or lower the activity level of services, security mechanisms and attacks by utilizing their competition for the limited resources such as central processing unit (CPU) time, memory and network bandwidth. Because all activities on a computer and network system share and compete for the limited system resources and the round-robin method of resource sharing is usually used on computers and networks, an increased level of one activity causes a decreased level of other activities sharing the resources. On one hand, this can mean a negative impact of attack activities on regular service and security activities as the increased level of attack activities reduce the level of service and security activities due to their sharing and competition for system resources. On the other hand, this sharing and competition for limited system resources can be used in a positive way for service and security activities. Instead of letting attack activities take more system resources, services and security mechanisms can increase their activity level when an attack occurs. The increased demand level of services and security mechanisms will allow them to increase their activity levels and obtain more system resources, thus suppressing the level of attack activities and sustaining QoS and QoIA levels during an attack. This general finding about QoS and QoIA adaptations is contradictory to what we commonly believe that attacks simply take away more system resources. System resources taken away by attacks can be taken back by regular services and security mechanisms by increasing the demand and thus activity level of services and security mechanisms for QoS and QoIA adaptations and survivability of services and security mechanisms running on a computer and network system under attacks.

TABLE OF CONTENTS

LIST OF FIGURE CAPTIONS

LIST OF TABLE CAPTIONS

Summary

As we increasingly rely on online services on computer and network systems to support critical operations in defense, banking, telecommunication, transportation, electronic power and many other domains, Quality of Service is of particular concern as it directly affects users' satisfaction and loyalty. QoS can be measured by different aspects of service performance such as throughput, delay, accuracy, security, and so on. From the perspective of resource management, competing service requests of users and their associated service activities change the state of system resources which in turn affects the QoS. Such dynamic relations of service activities, system state and QoS are the basis for determining and adapting service and system configurations to meet QoS requirements. However, models of such activity-state-quality relations are not readily available from the design of computer and network systems which provides mostly algorithm-based operational models. Activity-state-performance dynamic models from existing studies focus mostly on individual resources (e.g., router, CPU, memory, and hard disk) and limited aspects of dynamic models. There is a need for models capturing activity-state-quality dynamics at a more comprehensive, system scale including multiple resources, their interactions, their state changes with service activities, and their effects on the achieved QoS. Quality of Information Assurance (QoIA) is usually provided through security mechanisms such as encryption, intrusion detection, firewalls, etc.

Chapter 1 describes an empirical approach that is taken to uncover these cause-effect dynamics of service activities, system state and QoS and built activity-state-quality (ASQ) models. The empirical approach is applied to obtain ASQ models for a voice communication service. System dynamics data representing service activities, resource state and achieved QoS is collected from a real computer and network system setup under various service conditions. The experimental data is then analyzed to uncover service activities, system state and QoS relations and build activity-state-quality models. For the voice communication service, we uncover a number of state and QoS variables that are significantly affected by the three service activity parameters of the sampling rate, the number of clients and the buffer size, including *Committed Bytes_Memory*,

% Processor Time_Process, *IO Other Operations/sec_Process*, *Thread Count_Process*, *Page Faults/sec_Memory*, *Fragments Created/sec_IP*, and others. We also reveal four major categories of the ASQ relations: 1) a state or quality variable increases its values as the three service parameters of the sampling rate, the number of clients and the buffer size; 2) a state or quality variable increases its values as the sampling rate and the number of clients increases but decreases its values as the buffer size increases; 3) a state or quality variable increases its values as the number of clients increases with little effect of the sampling rate except at one low or high level, and first increases and then decreases its values as the buffer size increases, and 4) a state or quality variable decreases its values as the three service parameters increase. The ASQ relations provide us with insights into meeting higher demands of QoS with a higher sampling rate of voice data and more client requests by properly increasing the size of the buffer holding communication data before transmission over the network. The ASQ relations and the regression models defining the quantitative ASQ relations can be useful in predicting the change of the achieved QoS when initially configuring and later adapting the service activity parameters, the resource capacity and the service-resource binding to meet the QoS requirements. Although this study focuses on a communication-intensive voice communication service, the experimental and analytical methodology is applicable to investigating and developing dynamics models of service activity, system state and QoS cause-effect dynamics for other computer and network services.

In Chapter 2, the empirical approach is used to uncover dynamic effects and tradeoffs of service, security and attack activities for two service scenarios: a voice communication service with data encryption for QoIA and a motion detection service with intrusion detection for QoIA. System dynamics data representing service, security and attack activities, resource state and QoS is collected from both service scenarios. Statistical analysis on the experimental data uncovers dynamic effects and tradeoffs among services, security mechanism and cyber attacks. For each of the service scenarios, the uncovered system dynamics are grouped into categories. The characteristics of the relations in each category, tradeoff effects and their implications for QoS and QoIA adaptation, resource load balancing and system survivability are reported. Additionally, system dynamics variables showing unique attack effects for both service scenarios

were discovered. These variables can be used by the system to detect attacks. Some of these variables present unique characteristics to specific types of attacks. The insights gained by using our empirical approach enhance the understanding of the system dynamics and the interactions between services, security mechanisms and attacks.

Chapter 1 Models of Dynamic Relations among Service Activities, System State and QoS on Computer and Network Systems

1.1 Introduction

QoS has been essential for conventional, off-line service transactions, and has become increasingly important as many conventional service transactions are moved online with computer and network systems providing services. QoS of computer and network services can be measured by the performance of the service process in throughput, delay, and so on. Chen et al. (2004) describe QoS requirements of various network applications for online services, (e.g., web browsing, email, file transfer, audio and video broadcasting, audio and video on demand, audio and video conferencing, voice over IP, etc). QoS requirements for those online services are specified using metrics on timeliness (e.g., response time, delay and jitter), precision (e.g., bandwidth and loss rate) and accuracy of services (e.g., error rate). When competing service requests with specific QoS requirements come to a computer network system providing services, the system must determine if its limited system resources can satisfy these service requests at the required level of QoS and furthermore what service configuration, resource configuration and service-resource binding should be used for the achieved QoS (Jiang et al., 2008; Reisslein et al., 2002; Peng et al., 2002; Yang et al., 2008; Yau et al., 2007; Zhou et al., 2005; Ye, 2008). The competing service requests and resulting service activities change the state of limited system resources which in turn affects the achieved QoS (Ye, 2002). QoS is considered as a subset of service performance measures. For example, the increasing number of clients for a voice communication service is expected to produce more service activities which consume more system resources such as CPU, memory and network bandwidth and make these resources less available to each client, consequently leading to a decrease in the voice data throughput as a key measure of QoS for each client. Such dynamic relations of service activities, system state and QoS are the basis for determining and adapting service and system configurations to meet QoS requirements. However, models of such activity-state-quality relations are not readily available from the design of computer and network systems which provides mostly algorithm-based operational models. Activity-state-performance dynamic models from existing studies focus

mostly on individual resources (e.g., router, CPU, memory, and hard disk) and limited aspects of dynamic models. For examples, there are studies by

- Vazhkudai and Schopf (2002) on regression models for relations between disk load variation and file transfer time,
- Kapadia et al. (1999) on resource usage models in a computational grid environment,
- Ravindran and Hegazy (2001) on regression models between the timeliness performance of periodic tasks and external and internal load parameters such as CPU utilization,
- Shivam et al. (2006) on regression models of relations between varying assignments of computing, network and storage resources and application completion time, and
- Sun and Ifeachor (2006) on models of relations between the playout buffer control and the quality of voice over IP.

In the literature we did not find models of activity-state-quality dynamics during realistic operations of computer and network systems at a more comprehensive, system scale which account for a wide range of hardware and software resources (including CPU, memory, physical disk, caches, buffers, network interface, IP, TCP, UDP, Terminal Service, etc.), their interactions, their state changes with service activities, and their effects on the achieved QoS. Service and system configuration for QoS satisfaction should not simply consider the service effect on an individual resource or change the configuration of an individual resource because certain system resources (e.g., CPU and memory) often interact with and place constraints on one another. The achieved QoS depends on activity-state-quality dynamics at the system scale that takes into account effects of service activities on all system resources and QoS of all competing service processes/threads.

Due to the lack of models for activity-state-quality dynamics at a more comprehensive, system scale, existing studies on service and system configuration for QoS often bypass the issue of establishing models of activity-state-quality dynamic relations. Those studies focus only on the evaluation of QoS according to user-defined weights of QoS attributes without getting into models of realistic activity-state-quality dynamics to account for how the performance level of

various QoS attributes change dynamically with competing, dynamic service demands and the varying state, constraints and interactions of limited system resources. For example, Sha et al. (2009) presents a quality-based web services selection model that selects the best web service based on a set of QoS attributes that are related to performance, price, availability and latency. The model first selects a set of candidate web services that match functional requirements of user web service requests. Each candidate web service then receives a utility score of QoS based on a weighted sum of the normalized attribute values of QoS for the selection of the best web service. However, the study does not include information about how the attribute values of QoS on performance, availability and latency are obtained in real time and how the attribute values of QoS change dynamically with competing service requests and with state and constraints of system resources. There are other studies (Artaiam and Senivongse, 2008; Cao et al., 2009; Deng and Xing, 2009; Grah and Radcliffe, 2008; Kritikos and Plexousakis, 2007; Kulnarattana and Rongviriyapanish, 2009; Lv, 2009; Tran el al., 2009; Wang et al., 2006; Wu et al., 2007; Zhang et al., 2006; Zhang et al., 2009) on service and system configuration for meeting QoS requirements, all without the specification of activity-state-quality models of system dynamics or the basis for determining if and how service requests can be satisfied with the given state of limited system resources to achieve the required QoS.

The lack of models for realistic activity-state-quality dynamic relations at the system scale produces a significant gap in bringing existing work on service configuration and adaptation to real-world applications. The following sections presents our study to establish models of activity-state-quality dynamics models for a voice communication service as an illustration of how such models can be established empirically. System dynamics data representing service activities, resource state and achieved QoS is collected from a real computer and network system setup under various service conditions. The experimental data is then analyzed to uncover service activities, system state and QoS relations and build activity-state-quality models.

1.2 Methods, Assumptions, and Procedures

1.2.1 Cause-Effect Dynamics of Service Activities, Resource State and QoS

Figure 1 illustrates the cause-effect dynamics of service activities, resource state, and QoS in a typical user-process-resource interaction on computer and network systems (Ye, 2002). A service request from a user with QoS requirements calls for a service process which utilizes certain system resources and consequently changes the state of these resources. Changes of the resource state in turn affect the achieved QoS of the service process. We refer to models capturing such cause-effect dynamics of service activities (A), resource state (S) and QoS (Q) as Activity-State-Quality (ASQ) models.

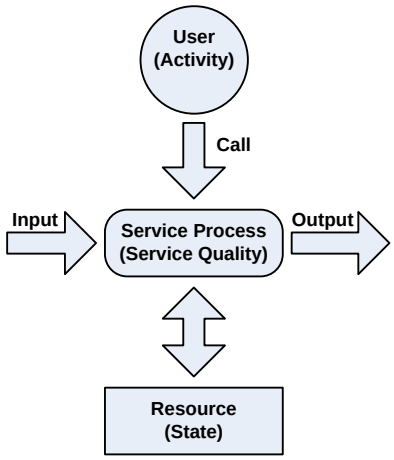


Figure 1: Cause-effect dynamics of service activities, resource state and QoS in the user-process-resource interaction.

1.2.2 Experiment Description: Voice Communication Service

Computers with a Windows operating system are used in our experiment. In the experiment, a voice communication service is set up in an online radio broadcasting context in which multiple clients simultaneously request and receive the voice communication service from a server which sends out real-time voice data streams of various quality levels controlled by the data sampling rate. Multiple clients run on their own computers with only one client on one computer. Figure 2 shows the computer and network setup with one server and five clients. Each computer has 1 GB

memory and Intel Pentium4 2.2 GHz processor. Both client and server computers are running Windows XP with SP2. The voice communication service is running on Internet Information Service 6.0. The voice communication service is developed by converting an open-source Video Conferencing software package (Abdel-qader, 2007) into a Web Service (WS) using C# in .NET Framework. The computer network system for the experiment stands alone without connections with any other computer and network system to avoid interferences.

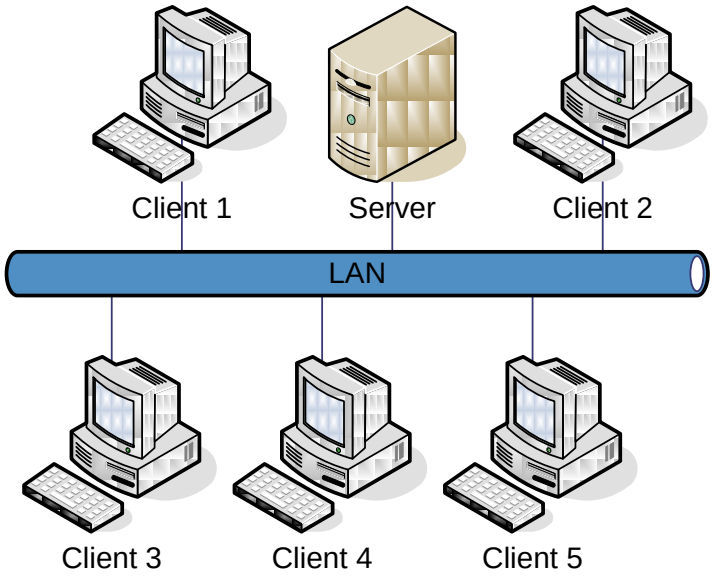


Figure 2: Computer and network setup for the voice communication service.

Windows operating systems provide Windows performance objects (Microsoft, 2003) which cover various aspects of process activities, resource state and service performance for many objects of resources and processes. Some examples of performance objects are Physical Disk, Memory, System, Process, Processor, IP, UDP, and TCP. Each object has a number of counters which reflect various aspects of activity, state and performance of the object. For example, the Memory object has a counter, *Available Bytes*, to show one aspect of the memory state. The IP object has a counter, *Fragmented Datagrams/sec*, to reflect the data transmission performance at the IP level. We use Windows performance objects to collect system dynamics data of various resources and processes which allow us to look into computer and network resources, their interactions, and their effects on QoS at the system scale.

To collect system dynamics data reflecting varying activity-state-quality dynamics for the voice communication service, we introduce various levels of service activities and system configurations in our experiment. The voice communication service is a communication-intensive application. QoS attributes for the voice communication service concern mainly the throughput and delay of voice data transmission. Hence, we want to select parameters of service activities and system configurations that are expected to affect the achieved QoS of the voice data communication service. Two parameters of service activities are selected for the experiment: 1) the sampling rate (Sa) which is used to record the voice data stream and thus determines the quality of voice data, and 2) the number of clients (C). These two parameters are used to reflect various levels of QoS demands and service activities in this voice communication service and to create competing service requests with QoS requirements that are expected to affect the achieved QoS of the voice communication service. The parameter of system configuration, the size of the buffer (B) on the server for storing the voice data before transmitting the data to a client over the network, is selected because the buffer size directly affects the throughput and delay of voice data transmission.

As shown in Table 1, each parameter has five levels in the experiment such that we collect data with sufficient granularity for data analysis to obtain ASQ relations and models. Hence, we have 125 (5 x 5 x 5) experimental conditions. The five levels of the sampling rate are denoted by Sa1, Sa2, Sa3, Sa4, and Sa5. The five levels of the number of clients are denoted by C1, C2, C3, C4, and C5. The five levels of the buffer size are denoted by B1, B2, B3, B4, and B5.

Table 1: Voice communication service experiment parameters and their levels

	Sampling Rate (Sa)	**Number of Clients (C)**	**Buffer Size (B)**
Level 1	44,100 Hz	1	16 Kbytes
Level 2	88,200 Hz	2	24 Kbytes
Level 3	132,300 Hz	3	32 Kbytes
Level 4	176,400 Hz	4	40 Kbytes
Level 5	220,500 Hz	5	48 Kbytes

ASQ models represent cause-effect relations among service activities (A), state of resources (S), and QoS (Q) of service processes. The two parameters of the sampling rate and the number of clients directly drive service activities and are considered as A variables in the ASQ models. The parameter of the buffer size affects the state of the buffer. With complex interactions of the buffer with other system resources during the process of the voice communication service, the parameter of the buffer size is also expected to affect the state of other system resources. Hence, the parameter of the buffer size is also considered as an A variable that is expected to affect the state of system resources and thus the achieved QoS.

System dynamics data of resource state and QoS of the voice communication service for S and Q variables in ASQ models are collected using eight Windows performance objects, including Physical Disk, Memory, System, Process, IP, UDP, TCP, and Server. These objects are selected because they are expected to be involved in the voice communication service. A number of variables collected through the eight Windows performance objects are related to the throughput and delay of voice data transmission—the QoS attributes of interest for the voice communication service. For example, *Fragments Created/sec_IP*, *Fragmented Datagrams/sec_IP*, *Datagrams Sent/sec_IP* and *Datagrams Sent/sec_UDP* are all related to the throughput of voice data transmission from the server to the clients. The variable *Fragments Created/sec_IP* is selected to measure the network throughput for the voice data transmission. The reasons for choosing *Fragments Created/sec_IP* over the other variables is provided in Section 1.3.2. The delay of voice data transmission includes 1) the time of generating and start sending voice data on the server which is closely related to the throughput of voice data transmission on the server, and 2) the transmission time of voice data over the network from the server to a client which is not directly measured and collected in our experiment. Hence, the delay of voice data transmission is not considered in this study. In our future research, we may also include additional parameters and variables such as the size of the receiving buffer, network delay and its variability, and a larger number of clients.

The experimental run under each of the 125 experimental conditions includes one minute of the voice communication service for a given level of the three service parameters. Sixty data observations under each experimental condition are collected with a sampling rate of 1 observation per second. The data is recorded in log files. Experimental data are collected on the server since the server data reflects the effect of multiple clients requesting the service. The analysis of the data is performed on the server data to obtain ASQ relations and models.

Some variables from Windows performance objects give accumulated values of system state or performance that increase over time, or change their values depending on the operation history. As a result, those variables are affected by the time when various service conditions are run. One example of such a variable is the *System Up Time* counter of the System object whose values increase over time. Hence, the time when a service condition is run has a confounding effect on those variables with various service conditions in the experiment. Such variables need to be removed from further data analysis that determines the effect of service conditions. To identify and remove such variables, we run a small-scale experiment involving only 3 of the 125 experimental conditions. The three experimental conditions in the small-scale experiment are the service condition with Sa1, C1 and B1, the service condition with Sa3, C3 and B3, and the service condition with Sa5, C5 and B5. The small-scale experiment includes two runs, each of which has three service conditions and two no-service conditions, as follows:

- Run 1: 1) no service at the beginning, 2) Sa1C1B1, 3) Sa3C3B3, 4) Sa5C5B5, and 5) no service at the end;
- Run 2 with the reversed order to that in Run 1: 1) no service, 2) Sa5C5B5, 3) Sa3C3B3, 4) Sa1C1B1, and 5) no service at the end.

The experimental data from these two runs with different orders of service conditions and no-service conditions allow us to analyze and examine possible confounding effects of service conditions and times when they are run on some variables and thus remove those variables. The remaining variables are kept for further data analysis to determine the effect of various service conditions using the data from the large-scale experiment with the full set of 125 service conditions.

1.2.3 Methodology of Data Analysis and Modeling

In this section, we first describe the steps of analyzing the data from the small-scale experiment to identify and remove variables whose values depend on the time when a service condition is run. Then we present the methodology of analyzing the data from the large-scale experiment to obtain the ASQ relations and models. The data modeling technique to build ASQ models is also described.

1.2.3.1 Small-Scale Experiment for Data Screening

The small-scale experiment includes two runs with different orders of three service conditions and two no-service conditions. For each run and each variable of the Windows performance objects data, we perform a Mann-Whitney test on the experimental data to compare each of the three service conditions (Sa1C1B1, Sa3C3B3, and Sa5C5B5) with each of the two no-service conditions (at the beginning and at the end). The Mann-Whitney test (Mann & Whitney, 1947a; Ye, 2008) is selected because the test uses a nonparametric statistic based on ranks and thus depends little on the probability density distribution of the data. Generally the Mann-Whitney test is as powerful as its typical parametric counterpart, the two-sample *t* test. The statistical software, Statistica7, is used to perform the Mann-Whitney test. Each Mann-Whitney test determines whether or not the effect of the service condition is significantly different from that of the no-service condition on a given variable. If there is a statistically significant difference, the specific difference (increase or decrease) of the service condition from the no-service condition is noted. Table 2 shows how the Mann-Whitney test results for each variable are recorded. Each cell in Table 2 has one of the three values: increase, decrease, or no significant difference.

For each variable, if the two rows of values for each run as recorded in Table 2 are not the same, the variable is removed. This is based on the consideration that the effect of each service condition on the variable should be the same when compared with the two no-service conditions at the beginning and at the end if the variable is not affected by the time and order of running a service condition and a no-service condition. If two rows of values for each run are the same but

they are different from two rows for another run, the variable is also removed because the difference indicates that the variable is affected by different orders of running service conditions or the time of running a given service condition. Only the remaining variables after the data screening are considered for further data analysis using the data of the large-scale experiment with the full set of 125 service conditions. Hence, the state and quality variables of the Windows performance objects in the next section refer to only the remaining variables after the data screening described in this section.

Table 2: Recording the Mann-Whitney test results for data screening using the data of the small-scale experiment.

Run	In Comparison with	Service Condition		
		Sa1C1B1	Sa3C3B3	Sa5C5B5
1	No-service at the beginning			
	No-service at the end			
2	No-service at the beginning			
	No-service at the end			

1.2.3.1 Large-Scale Experiment for Data Analysis and Modeling

The data from the large-scale experiment with the full set of 125 service conditions is used to uncover the relations of the service activity parameters with the resource state variables and QoS variables collected from the Windows performance objects. The following statistical data analyses are carried out.

1) A-SQ relation discovery and categorization. For each state or quality variable, the Analysis of Variance (ANOVA) in Statistica7 is performed with the three activity parameters of the sampling rate, the number of clients, and the buffer size (A's) as the independent variables and the state or quality variable (S or Q) as the dependent variable to determine the effects of A's on S or Q. For example, the increasing level of the sampling rate may cause an increase

in the used memory. If ANOVA results reveal a significant effect of one or more A variables on a S or Q variable, the Tukey's honest significant difference (HSD) test in Statistica7 is performed to determine how different levels of one or more A variables affect the S or Q variable. Then similar A-S or A-Q relations are grouped together and categorized into a certain type of A-SQ relations.

2) Development of the ASQ relation map. The representative A-S relations for each category of A-SQ relations from Step 1) are selected to construct the ASQ relation map which includes the three A variables, the selected S variables, and the Q variable measuring the network throughput of the voice communication service. In the ASQ relation map, each A, S, or Q variable is represented as a node. There is a directed link between an A variable and a Q variable to represent the A-S relation between this A variable and this Q variable. There is also a directed link between each S variable with the Q variable. For example, there may be a directed link between the sampling rate (an A variable) and the used memory (a S variable) and a directed link between the used memory (a S variable) and the network throughput variable (a Q variable).
3) ASQ modeling. For each S variable in the ASQ relations map, a regression model is built to give the quantitative relation of the S variable with one or more A variables. A regression model is also built to give the quantitative relation of the Q variable with the related S variables. We use the multiple regression procedure in Statistica to build a linear regression model if a linear regression model fits the data well; otherwise, a nonlinear regression technique such as the multivariate Adaptive Regression Splines (MARS) technique (Friedman, 1991) in the *earth* package of the *R* software (http://cran.r-project.org/web/packages/earth/earth.pdf) to build a nonlinear regression model.

1.3 Results and Discussion

This section describes and discusses the results of data screening and further data analysis and modeling. In the following text, a state or quality variable is denoted by *CounterName_ObjectName*.

1.3.1 Data Screening Results

The data screening steps described in Section 1.2.3.1 produce 106 remaining state and quality variables whose values are not affected by the time of running service conditions.

1.3.2 A-SQ Relations Categories

ANOVA for each of the remaining state and QoS variables from Section 1.3.1 reveal 28 S and Q variables that have a significant effect (with the *p*-value less than 0.05) of one or more A variables. For each significant effect on each S or Q variable, the Tukey's HSD test is performed to determine how different levels of one or more A variables affect the S or Q variable. For example, all the three A variables (the sampling rate, the number of clients, and the buffer size) have significant effects on the S variable, *%Committed Bytes in Use_Memory*, based on the ANOVA results. Table 3 gives the Tukey's test results for the effect of the sampling rate on *%Committed Bytes in Use_Memory*. Table 3 shows the groups of levels in the sampling rate that are significantly different from each other in their effects on *%Committed Bytes in Use_Memory*. If two levels marked by "***" fall into two different groups, there is a statistically significant difference between these two levels. For example, the Sa1 level of the sampling rate falls into group 1, whereas the Sa2 level of the sampling rate falls into group 2, indicating that the difference between Sa1 and Sa2 produces the statistically significant different effects on *%Committed Bytes in Use_Memory*. That is, the increase of the sampling rate from Sa1 to Sa2 significantly increases *%Committed Bytes in Use_Memory*. Table 3 indicates that *%Committed Bytes in Use_Memory* increases as the sampling rate increases. Tukey's test results for all the effects on *%Committed Bytes in Use_Memory* show that *%Committed Bytes in Use_Memory* increases as the sampling rate, the number of clients and the buffer size increase. Hence, *%Committed Bytes in Use_Memory* is categorized into a group of the S and Q variables with the A-SQ relations characterized as increasing with Sa, C and B (see Category 1 in Table 4).

Based on the Tukey's test results, the 28 S and Q variables, which show significant effects of one or more A variables based on the ANOVA results, are grouped into five categories of the A-SQ relations as shown in Table 4.

Table 3: Tukey's test results for the effect of the sampling rate on *%Committed Bytes in Use_Memory*

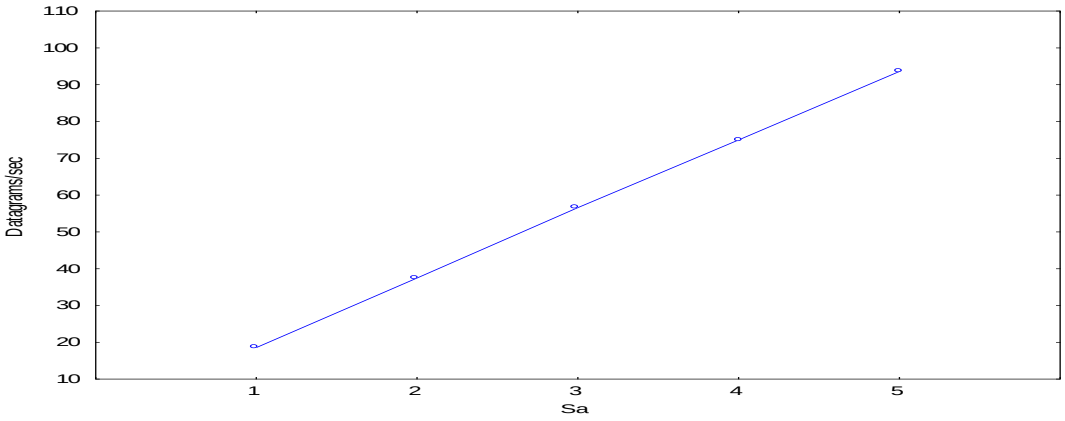
The Level of the Sampling Rate	The Average Value of *%Committed Bytes in Use_Memory*	Group				
		1	2	3	4	5
Sa1	11.77	***				
Sa2	12.09		***			
Sa3	12.40			***		
Sa4	12.64				***	
Sa5	12.91					***

Table 4: Five categories of A-SQ relations and the state and quality variables in each category

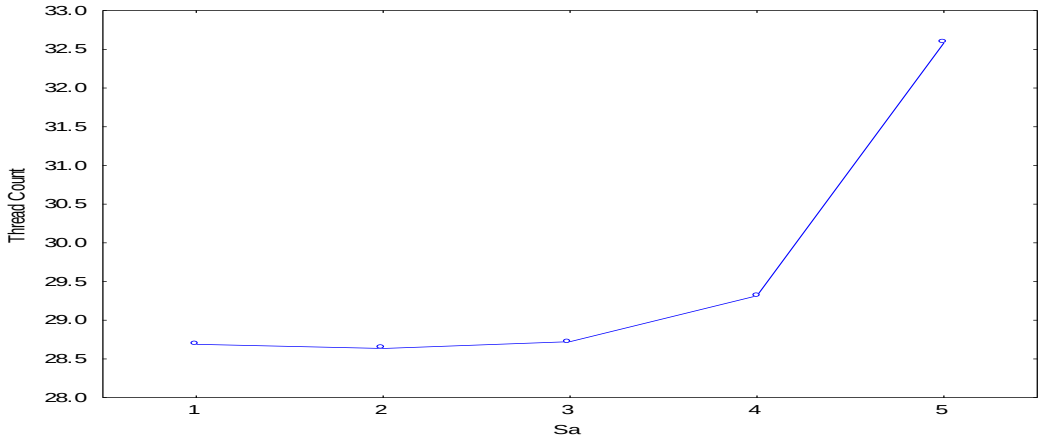
Category	**State and Quality Variables**
1. Increase with Sa and C and B	% Committed Bytes In Use_Memory, Committed Bytes_Memory
2. Increase with Sa and C, decrease with B	% Privileged Time_Process, % Processor Time_Process, % User Time_Process, Context Switches/sec_System, Datagrams/sec_UDP, Datagrams/sec_IP, Datagrams Sent/sec_UDP, Datagrams Sent/sec_IP, File Control Operations/sec_System, File Control Bytes/sec_System, Fragmented Datagrams/sec_IP, Fragments Created/sec_IP, IO Other Operations/sec_Process, IO Other Bytes/sec_Process
3. Increase with C, stable with Sa except at one end, inverse-U change with B	Thread Count_Process, Page Faults/sec_Memory
4. Decrease with Sa, C and B	Available Bytes_Memory, Available KBytes_Memory,

	Available MBytes_Memory
5. Inconsistent change with Sa, C and B and sometimes strong interaction of Sa, C and B	% Registry Quota In Use_System, Avg. Disk sec/Transfer_Physical Disk, Datagrams Received Delivered/sec_IP, Processor Queue Length_System, Datagrams Received/sec_IP, Datagrams Received/sec_UDP, Page Faults/sec_Process

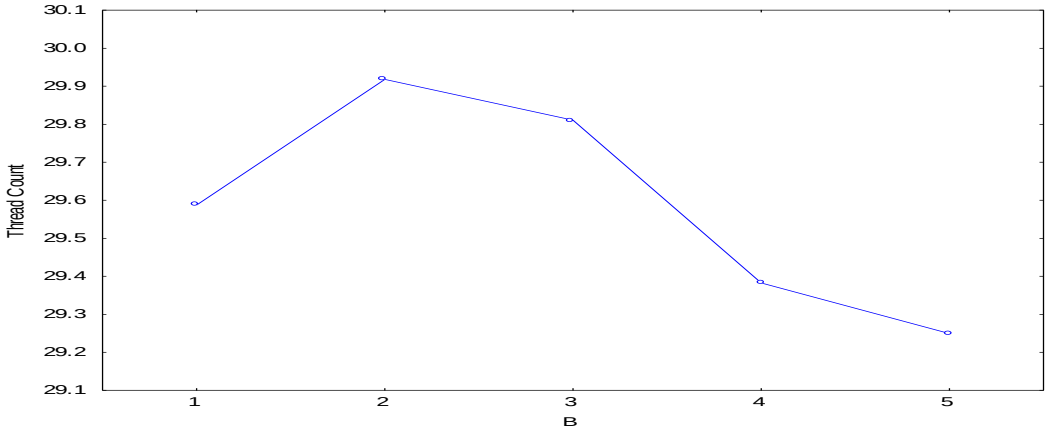
Category 1 has 2 state variables from the Memory object: *% Committed Bytes In Use_Memory* and *Committed Bytes_Memory*. These two state variables reflect the state of the memory concerning the memory consumption. The values of each S variable in this category increase as the sampling rate, the number of clients and the buffer size increase as shown in Figure 3a. Figure 3a plots the average values of the state variable, *Datagrams/sec_IP*, for the five levels of the sampling rate. The A-S relation of the state variables increasing with the three service parameters (the A variables) in this category indicates that the increasing level of the sampling rate, the number of clients and the buffer size in the voice communication service demands for more system memory. This A-S relation can be used to guide the adaptation of service and system configuration for the achieved QoS. When the higher level of QoS (e.g., a larger sampling rate for a better voice quality) is needed or more clients need to be served, the memory resource can be adapted by increasing the reserved memory, allocating more memory to the voice communication service, or through other means. The adaptation can be done through APIs, such as the *SetProcessWorkingSetSize* or *SetInformationJobObject* functions in Windows.



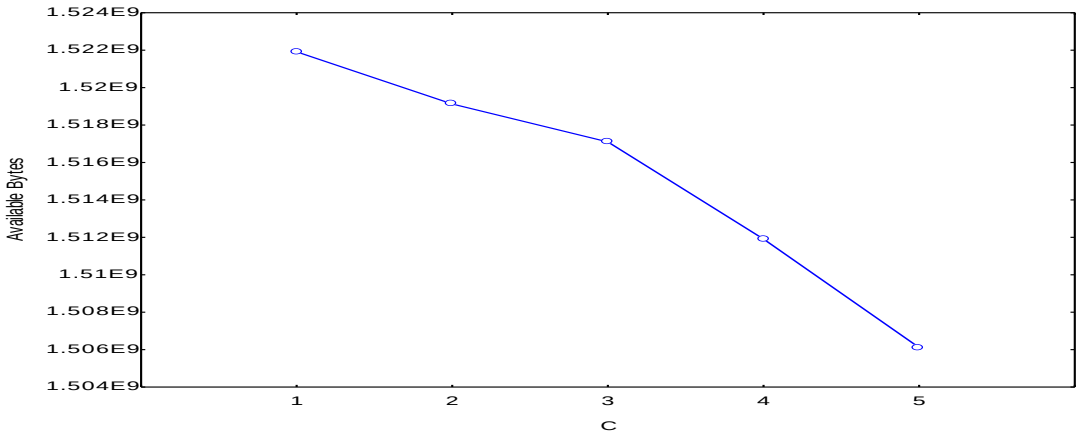
3a. "Increase with Sa" of *Datagrams/sec_IP* for 5 levels of the sampling rate.



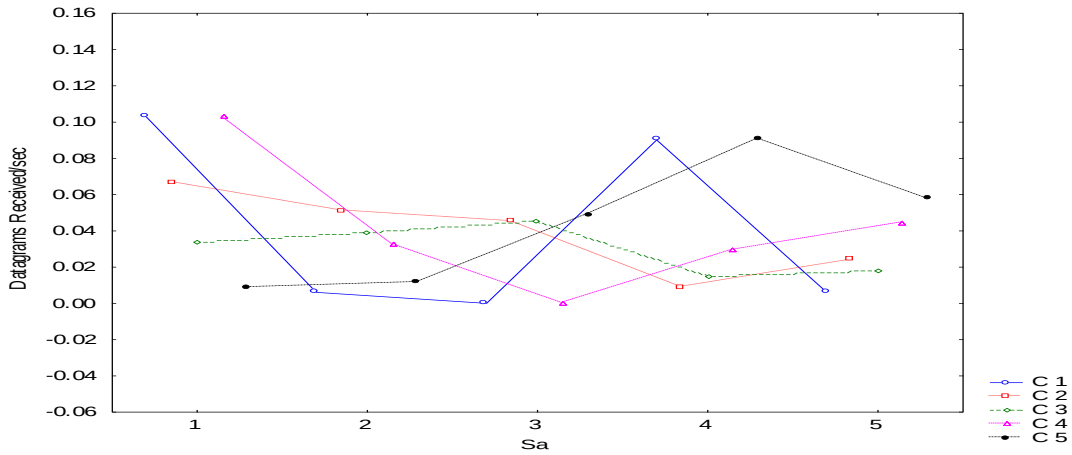
3b. "Stable with Sa except at one end" of *Thread Count_Process* for 5 levels of the sampling rate.



3c. "Inverse-U change with B" of *Thread Count_Process* for 5 levels of the buffer size.



3d. "Decrease with C" of *Available Bytes_Memory* for 5 levels of the number of clients.



3e. "Inconsistent change or strong interaction" of *Datagrams Received/sec_IP* for 25 levels of interactions between the sampling rate and the number of clients.

Figure 3: An illustration of relationships between activity variables and state-quality variables in the five A-SQ categories.

Category 2 includes 14 variables that reflect the state and/or performance of the Process, System, and network-related software objects concerning the use of the CPU and network interface resources at the hardware level. Specifically, the two variables from the UDP object (*Datagrams/sec_UDP* and *Datagrams Sent/sec_UDP*) and four variables from IP object (*Datagrams/sec_IP*, *Datagrams Sent/sec_IP*, *Fragmented Datagrams/sec_IP* and *Fragments Created/sec_IP*) reflect the amount of network communication generated mainly by the service

process of voice communication. Three variables from the Process object (*%Privileged Time_Process*, *%Processor Time_Process*, and *%User Time_Process*) reflect the CPU consumption by the service process of voice communication. The other two variables from the Process object (*IO Other Operations/sec_Process* and *IO Other Bytes/sec_Process*) reflect IO control operations by the service process of voice communication. The three variables from the System object (*Context Switches/sec_System*, *File Control Operations/sec_System* and *File Control Bytes/sec_System*) reflect the operating system (OS) operations for process scheduling and file system.

The values of each S or Q variable in category 2 increase as the sampling rate and the number of clients increase, but decrease as the buffer size increases as shown in Figure 3b. Hence, a higher level of QoS (i.e., a higher level of the sampling rate) and more user requests (i.e., a larger number of clients) demand for more CPU, communication bandwidth, and system IO and file resources. However, increasing the size of the communication buffer (B) can effectively reduce the number of datagrams sent by the voice communication service even with the similar amount of network data being sent, and subsequently reduce the workload on the CPU, system IO and file system. Hence, this category of A-SQ relations suggests a useful strategy for adapting service and system configuration by increasing the size of the communication buffer to maintain the workload level on CPU, system IO and file system resources and thus not to stress out these resources even when higher QoS requirements when more user requests (more clients) and better voice quality (higher sampling rate) are present.

Category 3 has two variables reflecting the state of the process and the memory, *Thread Count_Process* and *Page Faults/sec_Memory*. The values of each S variable in this category increase as the number of clients increases, show little change (stable) as the sampling rate increases except for the lowest or highest level (one end) of the sampling rate, and have an inverse-U change as the buffer size increases with higher values in the middle range of the buffer size (that is, the values of each variable increase and then decrease as the buffer size increases) as shown in Figure 3c. *Thread Count_Process* is determined by two factors: the number of clients

requesting voice data and the number of buffers waiting to be sent. As the number of clients increases, more threads need to be created to send voice data to the clients, resulting in the increase of the thread count. As the buffer size increases, the number of buffers filled up by the sampling thread (*NBF*) during a period of time decreases due to the larger buffer size. However, the number of buffers sent out by a sender thread (*NBS*) during a period of time also decreases due to the longer delay for sending the larger size of data. When *NBF* is larger than *NBS*, more sender threads need to be created to send out the voice data. When *NBF* is smaller than *NBS*, only one sender thread is needed for each client. Hence, as the buffer size increases, the difference between *NBF* and *NBS* first becomes larger causing the increase of *Thread Count_Process*, and then becomes smaller causing the decrease of *Thread Count_Process*. For *Page Faults/sec_Memory*, the increase of *Page Faults/sec_Memory* as the number of clients increases indicates that the competition on the system memory resource by the voice communication service and other applications/services in the system becomes more intensive when the number of clients increases. As the buffer size increases, more memory is consumed (as explained earlier for Category 1), which also leads to more intensive competition on the system memory resource. However, the number of memory accesses is reduced as the buffer size increases due to the reduced workload on the CPU and system IO (as explained earlier for Category 2), which leads to a smaller number of page faults. Hence, the value of *Page Faults/sec_Memory* first increases and then decreases as the buffer size increases. Since page faults cause slower memory accesses, it is always desirable to have less page faults. The A-S relation of *Thread Count_Process* and *Page Faults/sec_Memory* indicates that selecting a proper initial value of the buffer size is important for tuning the system workload imposed by the voice communication service.

Category 4 includes the three state variables from the Memory object measuring the same state of the memory in different units: *Available Bytes_Memory*, *Available Kbytes_Memory*, and *Available Mbytes_Memory*. The values of each variable in this category decrease as the sampling rate, the number of clients and the buffer size increase as shown in Figure 3d. The variables in this category measure the available memory, whereas the variables in Category 1 measure the

used memory consumption. Hence, the A-SQ relation in Category 4 conveys similar information to that in Category 1.

Category 5 has 7 state variables concerning the System, Physical Disk, IP, UDP, and Process objects. These variables measure the state and performance of those system and network resources which are not directly linked to the voice communication service. For example, the voice communication service generates mostly data sent out from the server rather than data received by the server as measured by *Datagrams Received Delivered/sec_IP*, *Datagrams Received/sec_IP*, and *Datagrams Received/sec_UDP* in Category 5. Each S variable in this category does not show a consistent change as the sampling rate, the number of clients and the buffer size increase. For example, the change of *Datagrams Received/sec*_IP with the sampling rate is different for different numbers of clients as shown in Figure 3e. The variables in this category are likely affected by not only the voice communication service but also system routine activities which together produce the inconsistent change pattern of these variables with the service activity parameters of the voice communication service. Hence, the variables in this category should not be considered as accurate measures of system state and QoS performance that are directly or solely linked to the voice communication service.

In summary, the 21 state and QoS variables in Categories 1-4 are directly related to the voice communication service, and show four major categories of the A-SQ relations.

1.3.3 ASQ Relation Map

Among the 21 state and quality variables from Section 1.3.2, some variables present similar information. While summarizing the ASQ relations into an ASQ relation map with each node representing a variable and a link representing a relation between two variables, we can keep only one variable among a group of variables that present similar information.

The two variables in Category 1 have the following relationship:

% Committed Bytes In Use_Memory = Committed Bytes_Memory / $Memory_{total}$,

where $Memory_{total}$ is the total size of system memory. Only *Committed Bytes_Memory* is kept in the ASQ relation map.

In category 2, *Datagrams/sec_IP*, *Datagrams/sec_UDP*, *Datagrams Sent/sec_IP*, *Datagrams Sent/sec_UDP*, *Fragmented Datagrams/sec_IP* and *Fragments Created/sec_IP* present similar information about the amount of network traffic created and sent out by the voice communication service at UDP and IP layers of the network protocols. Note that there is little incoming traffic to the voice communication server in our experiment. Since the sizes of datagrams depend on the communication buffer size used in our experiment, *Fragments Created/sec_IP* instead of *Datagrams Sent/sec_IP* closely reflects network throughput. Hence, *Fragments Created/sec_IP* is taken as the QoS measure of the network throughput for the voice communication service, and is the only variable among the six variables from the IP and UDP objects that is kept in the ASQ relation map. Also in category 2, *%Processor Time_Process*, *%User Time_Process* and *%Privileged Time_Process* has the following relation:

%Processor Time_Process = %User Time_Process + %Privileged Time_Process,

Only *%Processor Time_Process* is kept in the ASQ relation map. In category 2, *IO Other Operations/sec_Process* and *IO Other Bytes/sec_Process* of the Process Object presents similar information about the workload of IO devices or how busy the IO devices are. Only *IO Other Operations/sec_Process* is kept in the ASQ relation map. In category 2, *File Control Operations/sec_System* and *File Control Bytes/sec_System* of the System Object measure the control operations of the file system. Since IO devices are treated as special device files in Windows in order to simplify the system design, *File Control Operations/sec_System* includes both accesses to IO devices (already measured by *IO Other Operations/sec_System*) and regular files (mainly the log files in our experiments). In other words, *File Control Operations/sec_System* and *File Control Bytes/sec_System* are affected not only by the voice communication service activities by also the monitoring activities. Hence, these two variables are excluded in the ASQ relation map.

In category 3, both *Threat Count_Process* and *Page Faults_Memory* are kept since they give the state information for different system resources.

In category 4, *Available Bytes_Memory*, *Available Kbytes_Memory* and *Available Mbytes_Memory* of the Memory object present similar information to that by *Committed Bytes_Memory* of the Memory object which is kept in the ASQ relation map. Hence, the three variables in category 4 are excluded in the ASQ relation map.

In summary, the following five state variables:

- *Committed Bytes_Memory*
- *% Processor Time_Process*
- *IO Other Operations/sec_Process*
- *Thread Count_Process*
- *Page Faults/sec_Memory*,

and the following QoS variable:

- *Fragments Created/sec_IP*

are kept in the ASQ relation map. The results in Section 1.3.2 reveal the relations of the three service activity parameters, Sa, C and B, with these five state variables. The A-S relations are directly represented in the ASQ relation map as shown in Figure 4. As described in Section 1.2.1, we consider that the service activity parameters first act on the system resources and change the state of the system resources which in turn causes the change of process performance and thus the QoS variable. The voice communication service continuously samples voice data from the sound card of the system and transmits the sampled voice data to the clients. These operations require access to IO devices (mainly sound card and network interface) and creation of multiple threads, and consume CPU time and system memory. The states of CPU, memory, IO devices, and operating system have significant impact on the network throughput of the voice communication service, which is measured by *Fragments Created/sec_IP*. Hence, in the ASQ relation map shown in Figure 4, the relations of the five state variables with the QoS variable are represented.

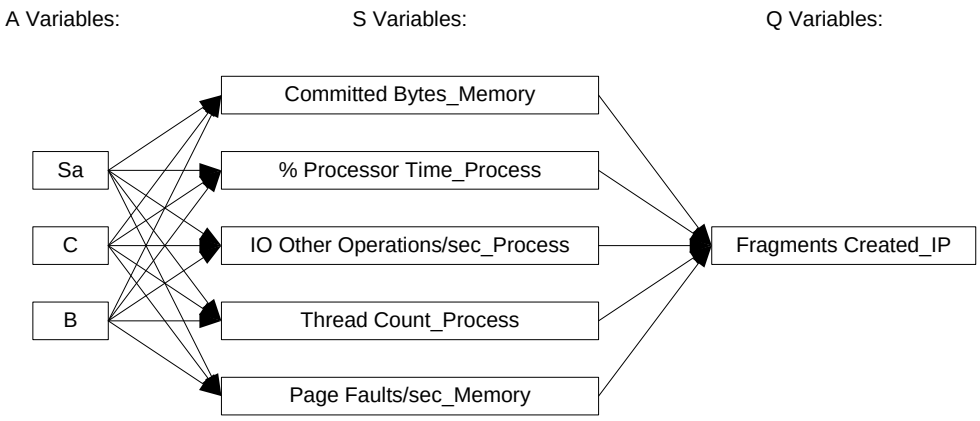


Figure 4: The ASQ relation map for the voice communication service

1.3.4 ASQ Models

For each state variable in the ASQ relation map, we obtain a linear regression model that represents the quantitative A-S relation of the three service activity parameters with the state variable. For the variable of the achieved QoS, we build a linear regression model to represent the quantitative S-Q relation of the five state variables with the variable of the achieved QoS. These regression models are presented in Table 5. In these regression models:

C = 1, 2, 3, 4, 5 for one to five clients, respectively,

S = 1, 2, 3, 4, or 5 for 44100, 88200, 132300, 176400, 220500 Hz, respectively,

B = 1, 2, 3, 4, or 5 for 16384, 24576, 32768, 40960, 49152 Kbytes, respectively.

Since the difference between two successive levels of S (e.g., 88200-44100=132300-88200) and B (24675-16384=32768-24576) is the same, S = 1, 2, 3, 4 or 5 and C = 1, 2, 3, 4, or 5 in the regression models is simply a scaled value for the original value of S and B. The R-square value for each regression model gives the goodness-of-fit of the regression model to the data with a larger value indicating a better fit. For *Page Faults/sec_Memory*, the linear regression model does not fit the data well. Hence, we use the MARS technique to build a nonlinear regression model with the R^2 value of 0.8160. The MARS model is not presented in the paper due to its complex form.

Table 5: Linear regression models for A-S and S-Q relations

S or Q Variable	Regression Model	R^2
Committed Bytes_Memory (S_1)	$S_1 = 429097992 + 11746708(S) + 15190725(C) + 426122(B)$	.9939
%Processor Time_Process (S_2)	$S_2 = -2.30198 + 1.02864(S) + 0.87906(C) - 0.33787(B)$	.7569
IO Other Operations/sec_Process (S_3)	$S_3 = -19.7548 + 37.5533(S) + 37.4458(C) – 30.9270(B)$	.7986
Thread Count_Process (S_4)	$S_4 = 14.79020 + 0.84680(S) + 4.20747(C) - 0.12113(B)$	.9595
Page Faults/sec_Memory (S_5)	$S_{15} = – 119.014 + 24.065(S) + 199.893(C) + 4.758(B)$	.5160
Fragments Created/sec_IP (Q)	$Q = – 7544.65 + 0(S_1) + 174.76(S_2) – 7.86(S_3) – 0.06(S_4)$	.9419

1.4 Conclusions

We present our methodology of data collection, data analysis and data modeling to establish activity-state-quality models which can be used to enable the configuration and adaptation of services and system resources for meeting QoS requirements. We illustrate the methodology using the voice communication service. For the voice communication service, we uncover a number of state and QoS variables that are significantly affected by the three service activity parameters of the sampling rate, the number of clients and the buffer size, including *Committed Bytes_Memory*, *% Processor Time_Process*, *IO Other Operations/sec_Process*, *Thread Count_Process*, *Page Faults/sec_Memory*, *Fragments Created/sec_IP*, and others. We also reveal four major categories of the ASQ relations: 1) a state or quality variable increases its values as the three service parameters of the sampling rate, the number of clients and the buffer size; 2) a state or quality variable increases its values as the sampling rate and the number of clients increases but decreases its values as the buffer size increases; 3) a state or quality variable increases its values as the number of clients increases with little effect of the sampling rate except at one low or high level, and first increases and then decreases its values as the buffer size increases, and 4) a state or quality variable decreases its values as the three service parameters increase. The ASQ relations provide us with insights into meeting higher demands of QoS with a higher sampling rate of voice data and more client requests by properly increasing the size of the

buffer holding communication data before transmission over the network. The ASQ relations and the regression models defining the quantitative ASQ relations can be useful in predicting the change of the achieved QoS when initially configuring and later adapting the service activity parameters, the resource capacity and the service-resource binding to meet the QoS requirements. Although this study focuses on a communication-intensive voice communication service, the experimental and analytical methodology is applicable to investigating and developing dynamics models of service activity, system state and QoS cause-effect dynamics for other computer and network services.

References

Abdel-qader, F. M. (2007). *Examples to create your conferencing system in .NET, C# VOIP & video conferencing systems using H.323 and TAPI 3.* Retrieved 03/16, 2009, from http://www.codeproject.com/KB/IP/Video_Voice_Conferencing.aspx?fid=373359&df=90&mpp=25&noise=3&sort=Position&view=Quick&select=2645686

Artaiam, N., Senivongse, T. (2008). Enhancing Service-Side QoS Monitoring for Web Services. *9th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing,* 765-770.

Cao, J., Huang J., Wang, G., Gu, J. (2009). QoS and Preference based Web Service Evaluation Approach. *8th International Conference on Grid and Cooperative Computing,* 420-426.

Chen, Y., Farley, T., & Ye, N. (2004). QoS requirements of network applications on the internet. *Inf.Knowl.Syst.Manag.,* 4(1), 55-76.

Deng, X., Xing, C. A (2009). QoS-oriented Optimization Model for Web Service Group. 8th *IEEE/ACIS Conference on Computer and Information Science,* 903-909.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The Annals of Statistics,* 19(1), 1-67.

Grah, M., Radcliffe, P. (2008). Dynamic QoS and Network Control for Commercial VoIP Systems in Future Heterogeneous Networks. *10th International Symposium on Multimedia,* 356-383.

Jiang, C., Hu, H., Cai, K., Huang, D., & Yau, S. S. (2008). An intelligent control architecture for adaptive service-based software systems with workflow patterns. *Computer Software and Applications Conference, Annual International,* 824-829.

Kapadia, H. N., Fortes, A. B. J., & Brodley, C. E, (1999). Predictive Application-Performance Modeling in a Computational Grid Environment, *Proceedings of the 8th IEEE International Symposium on High Performance Distributed Computing*.

Kritikos, K., Plexousakis, D. (2007). Requirements for QoS-based Web Service Description and

Discovery. *31st Annual International Computer Software and Applications Conference (COMPSAC)*, 467-472.

Kulnarattana, L., Rongviriyapanish, S. (2009). A Client Perceived QoS Model for Web Services Selection. *6th International Conference on Electrical Engineers/Electronics, Computer, Telecommunications and Information Technology*, 731-734.

Lv, T. (2009). Research on Workflow QoS. *International Joint Conference on Artificial Intelligence*, 219-221.

Mann, H. B., & Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics,* 18(1), 50-60.

Microsoft. (2003). *Window server 2003 performance counters reference*. Retrieved 03/01, 2009, from http://technet2.microsoft.com.ezproxy1.lib.asu.edu/windowsserver/en/library/3fb01419-b1ab-4f52-a9f8-09d5ebeb9ef21033.mspx

Peng X., Telkamp, T., Fineberg, V., Cheng C., & Ni, L. M. (2002). A practical approach for providing QoS in the internet backbone. *IEEE Communications Magazine,* 40(12), 56-62.

Ravindran, B., Hegazy, T. (2001). A Predictive Algorithm for Adaptive Resource Management of Periodic Tasks in Asynchronous Real-Time Distributed Systems in *International Parallel and Distributed Processing Symposium*.

Reisslein, M., Ross, K. W., & Rajagopal, S. (2002). A framework for guaranteeing statistical QoS. *Networking, IEEE/ACM Transactions on,* 10(1), 27-42.

Sha, L., Shaozhong, G., Xin, C., Mingjing, L. (2009). A QoS based Web Service Selection model. *2009 International Forum on Information Technology and Applications,* 353-356.

Shivam, P., Babu, S. & Chase, J.S., (2006). Learning Application Models for Utility Resource Planning, Autonomic Computing. *ICAC '06. IEEE International Conference on* , 255-264.

Sun, L. & Ifeachor, E.C. (2006). Voice quality prediction models and their application in VoIP networks, *IEEE Transactions on Multimedia*, 8(4), 809 – 820.

Tran, V. X., Tsuji, H., Masuda, R. (2009). A new QoS ontology and its QoS-based ranking algorithm for Web Services. *Simulation Modelling Practice and Theory,* vol. 17, 1378-1379.

Vazhkudai, S., Schopf, J. M., (2002). Using Disk Throughput Data in Predictions of End-to-End

Grid Data Transfers, *Grid Computing, LNCS*, 291-304.

Wang, X., Vitvar, T., Kerrigan, M., Toma, I. (2006). A QoS-aware Selection Model for Semantic Web Services. *4th International Conference on Service Oriented Computing, 390-401.*

Wu, B., Chi, C., Xu, S. (2007). Service Selection Model based on QoS Reference Vector. *2007 IEEE Congress on Service*, 270-277.

Yang, Z., Ye, N., & Lai, Y. C. (2008). QoS model of a router with feedback control. *Quality and Reliability Engineering Int'l,* 22(4), 429-444.

Yau, S. S., Huang, D., Zhu L., & Cai, K. (2007). A software cybernetics approach to deploying and scheduling. *FTDCS '07: Proceedings of the 11th IEEE International Workshop on Future Trends of Distributed Computing Systems,* 149-156.

Ye, N. (2002). QoS-centric stateful resource management in information systems. *Information Systems Frontiers,* 4(2), 149-160.

Ye, N. (2008). *Secure computer and network systems: Modeling, analysis and desig*n Wiley Publishing: London, UK.

Zhang, B., Shi, Y., Chen, Y. (2006). A Policy-based Adaptation Method for Service Composition. *1st International Symposium on Pervasive Composition,* 619-624.

Zhang, G., Zhang, H., Wang, Z. (2009). A QoS-based web services selection method for dynamic web service composition. *First International Workshop on Education Technology and Computer Science,* 832-835.

Zhou, J., Cooper, K., Yen, I., & Paul, R. (2005). Rule-base technique for component adaptation to support QoS-based reconfiguration. *Object-Oriented Real-Time Distributed Computing, IEEE International Symposium,* 426-433.

List of Abbreviations and Acronyms

A	Service **A**ctivity
ANOVA	**An**alysis **o**f **Va**riance
A-S	Service **A**ctivities to Resource **S**tate
A-SQ	Service **A**ctivities to Resource **S**tate/**Q**oS Metric
ASQ	**A**ctivity-**S**tate-**Q**uality
B	**B**uffer Size
C	Number of **C**lients
CPU	**C**entral **P**rocessing **U**nit
HSD	**H**onest **S**ignificant **D**ifference
IO	**I**nput/**O**utput
IP	**I**nternet **P**rotocol
MARS	**M**ultivariate **A**daptive **R**egression **S**plines
NBF	**N**umber of **B**uffers **F**illed
NBS	**N**umber of **B**uffers **S**ent
OS	**O**perating **S**ystem
Q	**Q**oS Metric
QoS	**Q**uality **o**f **S**ervice
QoIA	**Q**uality **o**f **I**nformation **A**ssurance
S	Resource **S**tate
Sa	**Sa**mpling rate
S-Q	**S**tate of resources to **Q**oS Metric
TCP	**T**ransmission **C**ontrol **P**rotocol
UDP	**U**ser **D**atagram **P**rotocol
WS	**W**eb **S**ervice

Chapter 2 Dynamic Effects and Tradeoffs of Service, Security and Attack Activities

2.1 Introduction

When cyber attacks cause unforeseen damage to a cyber system, the system must have the capability of dynamically adapting itself to survive and recover from the damage (Lipson and Fisher, 1999; Atighetchi et al., 2004; Yi and Zhang, 2005; Zhang et al., 2007; Xiao et al., 2007; Zuo and Panda, 2009).The dynamic adaptation usually involves pulling out reserves of additional system resources or making tradeoffs within limitations of existing system resources. The use of system reserves for infrequent occurrence of system damage by cyber attacks is costly. There is always a limitation in how much system reserves can be held. There is always a possibility that the damage caused by cyber attacks leads to a severe shortage of system resources. Hence, tradeoffs within the limitation of given system resources must be made for the system to sustain its more critical services under the damage of cyber attacks.

Three types of activities may occur on a computer and network system at the same time: service activities, security activities and cyber attack activities. Various services may be running on the system to fulfill a mission. Security mechanisms may also be running at the same time to protect the system from cyber attacks. Cyber attacks may be additional activities on the system launched by malicious users with the purpose to compromise system resources, services and security. Any of these activities cause changes in the state (e.g., availability) of system resources which in turn lead to changes in system performance and Quality of Service (QoS) (Ye, 2002; Ye, Newman and Farley, 2005; Ye, 2008). Understanding such cause-effect relationships of activities, system state and QoS are essential to uncover the dynamic effects and tradeoffs among services, security and attack activities required for QoS adaptation, system survivability and load balancing. However, as stated in Section 1.1 such models of activity-state-QoS (ASQ) dynamics are not readily available from the design of computer and network systems which provides mostly algorithm-based operational models. Activity-state-performance dynamic models from existing studies focus mostly on individual resources (e.g., router, CPU, memory, and hard disk) and

limited aspects of dynamic models. There is a current need for models capturing activity-state-quality dynamics at a more comprehensive, system scale which consider a wide range of hardware and software resources (including CPU, memory, physical disk, caches, buffers, network interface, IP, TCP, UDP, Terminal Service, etc.), their interactions, their state changes with service, security and attack activities, and their effects on the achieved QoS.

The dynamic adaptation for survivability should not simply consider the effects of services, security mechanisms and cyber attacks on individual resources or the reconfiguration of an individual resource because all system resources interact with and place constraints on one another. The QoS depends on activity-state-QoS dynamics at the system scale, i.e., effects of service, security and attack activities on all system resources and QoS of all competing service processes/threads.

Section 2.2 describes our methodology for uncovering dynamic effects and tradeoffs of service, security and attack activities required for QoS adaptation, system survivability and load balancing. The methodology includes an experiment to collect system dynamics data under various conditions of services, security mechanisms and cyber attacks and the statistical analysis of the experimental data to uncover the dynamic effects and tradeoffs among services, security mechanism and cyber attacks. The methodology is illustrated through a specific experiment involving two specific services of voice data communication and motion detection, two security mechanism of data encryption and intrusion detection, and five cyber attacks of address resolution protocol (ARP) Poison, ping flood, vulnerability scan, fork bomb, and remote dictionary. The dynamic effects and tradeoffs uncovered for the experimental conditions and their implications in terms of QoS adaptation, survivability and load balancing are presented in Section 2.3. Section 2.4 presents the key conclusions arising for this part of the research.

2. 2 Methods, Assumptions, and Procedures

In this section, the methodology for uncovering dynamic effects and tradeoffs of service, security and attack activities is described. The experiments are designed to capture the Activity-State-

Quality dynamics of service, security and attack activities. The data collected from the experiments will be analyzed to uncover dynamic effects and tradeoffs required for QoS adaptation, load balancing and system survivability.

The experiments involves two specific services (voice communication and motion detection), two security mechanisms (data encryption and intrusion detection), and five cyber attacks (ARP Poison, ping flood, vulnerability scan, fork bomb, and remote dictionary).

The voice communication was previously described in 1.2.2. The motion detection is a computational-intensive service with delay and motion level as main QoS measures. The data encryption service uses a well-know and efficient symmetric-key encryption algorithm called Advanced Encryption Standard (Daemen and Rijmen, 2001). Snort is one of the most widely deployed network intrusion detection and prevention systems (Sourcefire, Inc.). A complete list and description of attack mechanisms can be found at the "Mechanisms of Attack" website (Mitre Corporation). In this project, ARP poison is a man in the middle type of attack that poisons the ARP tables of the victim computers. Cain and Abel® v4.9.30 (Montoro) is selected to perform the ARP Poison attack. Ping flood is a denial of service (DOS) type of attack that mainly compromises network bandwidth. Ping ® v2.0 (Tools4ever) is selected to perform the attack. Vulnerability scan is an analytic type of attack that search for system vulnerabilities through the network open ports. Nmap ® v4.76 (Insecure.Org) is selected to perform the attack. Fork Bomb is a resource depletion type of attack that creates processes/threads in the victim computer to compromise the availability of system resources to legitimate services (Ye, 2008). Remote dictionary is a dictionary-based type of attack used to gain access to the administrator account via Windows desktop connection utility. Tscrack® v2.1 (Ye, 2008) is selected to perform the remote dictionary attack. Services and security mechanisms are combined into two experiments: an encrypted voice communication service and a network-secured motion detection service.

2.2.1 Experiment Description: Encrypted Voice Communication Service

An experiment is designed to investigate tradeoffs effects among a voice communication service (VCS), a security service (SS) for data encryption, and several attacks including ARP Poison, fork bomb, NMAP, ping, and remote dictionary. The voice communication service has three service parameters: the sampling rate (Sa) which determines the quality of the sampled voice data, the number of clients (C) who request the voice communication service, and the size of the buffer (B) holding the sampled voice data at the server of the voice communication service before transmitting it over the network to a client. SS has two service parameters: the key length (K) for encryption, and the encryption percentage (E). Table 6 shows the levels of the parameters used in the experiments. 243 conditions (3*3*3*3*3 combinations of levels for Sa, C, B, E, and K) of the experiments are run while each of the five attacks is executed and also while no attack is executed. Table 6 shows totally 486 (=3*3*3*3*3*2) combinations of experimental conditions for the encrypted voice communication service. The 486 conditions are run in a random order with one run for each condition to collect 30 observations of system dynamics data from the server. Then the reverse order of the random order is used to run the 486 conditions again. One data observation is collected every second. System dynamics data from both orders of experimental runs are used in data analysis in order to eliminate the possible effect of the order for running the experimental conditions on data observations.

Windows performance objects (Microsoft, 2003) are used to collect system dynamics data that reflects activities, state of system resources, and QoS performance on the server. In total 15 performance objects, including Process, Memory, Paging File, Physical Disk, IP, UDP, TCP, Server, System, Web Services, Network, Objects, Processor, Redirector and Terminal Service session (TSS), are used to collect data. Each object has a number of counters that give activity, state and performance variables of the object. There are 384 variables from these 15 performance objects.

The experimental data is first screened to remove variables that are not related to effects of the experimental parameters. The remaining variables are then analyzed to uncover activity-state-QoS dynamics and tradeoff effects of the voice communication service, the security service and cyber attacks.

Table 6: Encrypted voice communication experiment parameters and their levels

Parameters	**Level 1**	**Level 2**	**Level 3**
Sampling rate (Sa)	44100Hz	132300Hz	220500Hz
Number of Clients (C)	1	3	5
Buffer size (B)	16Kbytes	32KBytes	48Kbytes
Encryption Percentage (E)	0%	50%	100%
Key Length (K)	128 bits	192 bits	256 bits
Cyber Attack (At)	no attack	attack	

Data variables are removed from further data analysis if they are not related to effects of the experimental parameters. A data variable is removed for one of the following reasons.

- The variable records the highest or peak value since the computer is last restarted. For example, the variable *Virtual bytes peak* of the Process object records the highest virtual address space in bytes used by the voice communication service since the server is last restarted. Hence, in one order of the experimental runs, values of the variable keep increasing over time.
- The variable collects the cumulative value over time since the computer is last restarted. For example, the variable *Datagrams outbound discarded* of the IP object counts the number of output IP datagrams with no errors that are discarded due to reasons such as lack of buffer space since the computer is last restarted. Values of the variable keep increasing with time.
- The variable collects data that is not affected by the experimental conditions. For example, the variable *Priority base* of the Process object measures the base priority of the voice communication service running. Neither of the services is designed to adjust the service priority. Totally 165 variables are removed. The remaining 219 variables are used in further data analysis. An Analysis of Variance (ANOVA) is performed for each variable of

Windows performance object to determine if the variable shows effects of the experimental parameters in Table 6. These experimental parameters are the independent variables in the ANOVA. The variable from the Windows performance object is the dependent variable in the ANOVA. The Tukey's Honest Significant Difference (HSD) test is also performed on each of the variables with a significant main or interaction effect of the experimental parameters to determine experimental conditions that are significantly different from one another. Statistica 7® is used to perform ANOVA and Tukey's HSD tests.

2.2.2 Experiment Description: Network-secured Motion Detection Service

An experiment is designed to investigate tradeoffs effects among a motion detection service (MDS), an intrusion detection mechanism (ID) and several attacks including ARP Poison, fork bomb, NMAP, ping, and remote dictionary. The Network-secured motion detection experiment was setup on an isolated network environment as shown in Figure 5. Each computer has 1 GB memory and Intel Pentium4 2.2 GHz processor. Both client and server computers are running Windows XP with SP2. The motion detection service is running on Internet Information Service 6.0. The motion detection service is developed by converting an open-source motion detection software package (Kirillov 2007) into a Web Service using C# in .NET Framework.

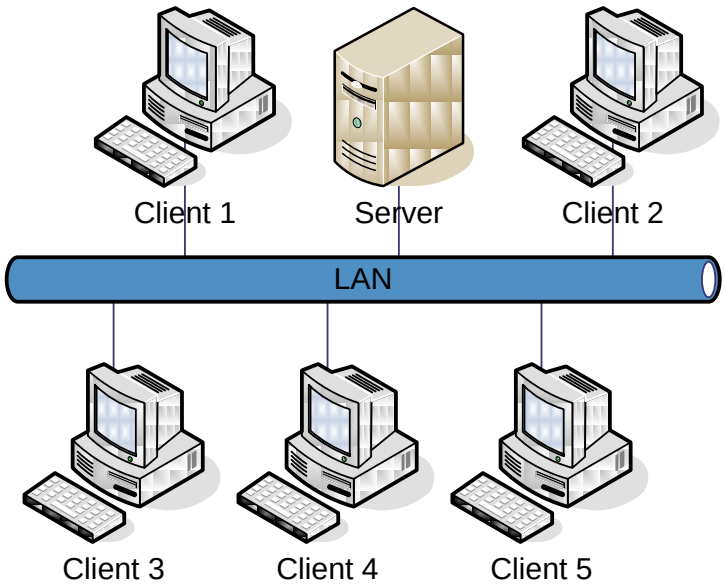


Figure 5 : The computer and network setup for the network-secured motion detection experiment.

The motion detection service is a computation-intensive service. From a user point of view, the major QoS attributes of concern are the processing delay of the motion detection service (i.e. the delay of processing each frame from a video stream) and the accuracy of detection results. Since our goal is not to evaluate and compare different motion detection algorithms, we did not consider the usage of different algorithms as another service parameter, i.e. only one algorithm was used in the experiment. For simplicity, we used a simple algorithm, which first extracts a background frame from the initial frames of a video stream, and then calculate the differences between the subsequent frames and the background frame. Such differences are recorded as the motion levels, which later can be used to measure the accuracy of the motion detection service. The formula of calculating motion levels is as follows: *Motion level = Number of changed pixels / Total number of pixels.* Multiple clients may simultaneously request the server to process a surveillance video stream with a specified resolution to detect whether there is any motion. Each client will be running on one computer. In order to have more comparable and repeatable experiment results, pre-recorded videos with different resolutions were used instead of real-time video streams from peripheral devices. When a client request is received, a video file with the specified resolution will be opened and processed frame-by-frame by the MDS at the rate of 20 frames per second (to simulate real-time video streams from peripheral devices such as a webcam).

Two service parameters are considered: the video resolution and the number of concurrent users (clients). ID is run independently from MDS as the security protection mechanism. Table 7 show the levels of the parameters used in the experiments.

Table 7: Network-secured motion detection experiment parameters and their levels

Parameter	**Level 1**	**Level 2**	**Level 3**
Video resolution (R) (pix. X pix.)	22 x 18	44 x 36	88 x 72
Number of threads/clients (T)	1	3	5
Intrusion Detection (ID)	no ID	ID	
Cyber Attack (At)	no attack	attack	

Table 7 shows totally 36 (=3*3*2*2) combinations of experimental conditions for the network-

secured motion detection service. The 36 conditions are run in a random order with one run for each condition to collect 30 observations of system dynamics data from the server. Then the reverse order of the random order is used to run the 36 conditions again. One data observation is collected every second. System dynamics data from both orders of experimental runs are used in data analysis in order to eliminate the possible effect of the order for running the experimental conditions on data observations. Windows performance objects (Microsoft, 2003) are used to collect system dynamics data that reflects activities, state of system resources, and QoS performance on the server. In total 15 performance objects, including Process, Memory, Paging File, Physical Disk, IP, UDP, TCP, Server, System, Web Services, Network, Objects, Processor, Redirector and Terminal Service session (TSS), are used to collect data. Each object has a number of counters that give activity, state and performance variables of the object. There are 384 variables from these 15 performance objects.

Similar to the analysis for the encrypted voice communication service, the experimental data collected during the network-secured motion detections service is first screened to remove variables that are not related to effects of the experimental parameters. Totally 165 variables are removed. The remaining 219 variables are used in further data analysis. An Analysis of Variance (ANOVA) is performed for each variable of Windows performance object to determine if the variable shows effects of the experimental parameters in Table 7. These experimental parameters are the independent variables in the ANOVA. The variable from the Windows performance object is the dependent variable in the ANOVA. The Tukey's HSD (Honest Significant Difference) test is also performed on each of the variables with a significant main or interaction effect of the experimental parameters to determine experimental conditions that are significantly different from one another. Statistica 7® is used to perform ANOVA and Tukey's HSD test.

2.3 Results and Discussion

2.3.1 Dynamic effects and Tradeoffs of the Voice Communication Service, the Data Encryption Service and Attacks

Table 8 shows the types of relations and tradeoff effects of the experimental parameters with system dynamics variables for the voice communication service, the data encryption service, and the cyber attacks. *Note: the variables have the V relation with respect to the parameters (Sa, C, B, E and K) if they first decrease their values as they change from level 1 to level 2 and then increase their values as they change from level 2 to level 3. The variables have the inverse-V (Λ) relation with respect to the parameters (Sa, C, B, E and K) if they first increase their values as they change from level 1 to level 2 but then decrease their values as they change from level 2 to level 3. Levels 1, 2 and 3 are defined in table 6.*

Table 8: System dynamics for the voice communication service, the data encryption service, and the cyber attacks: Categories

Category	Characteristics of Relations and Tradeoff Effects	Object	Variables
1	At↓E↑Sa↑C↑(K↓): • 17 variables in total • All variables decrease with At. • All variables increase with E, Sa and C. • 14 variables (all Process, System and Web Service variables) decrease with K.	**Process** (9 variables)	*Activity variables:* Page Faults/sec, IO Read Operations/sec, IO Write Operations/sec, IO Data Operations/sec, IO Other Operations/sec, IO Read Bytes/sec, IO Write Bytes/sec, IO Data Bytes/sec *State variable:* %Privileged Time
		System (3 variables)	*Activity variable:* File Read Bytes/sec, File Write Bytes/sec, File Control Bytes/sec
		Physical Disk (2 variables)	*Activity variables:* Avg. Disk Bytes/Transfer, Avg. Disk Bytes/Write
		Web Service (2 variables)	*Activity variables:* Post Requests/sec, ISAPI Extension Requests/sec
		Terminal Service Session (1 variable)	*Activity variable:* Page Faults/sec
2	At↓EΛ(Sa↑C↑): • 2 variables in total • All variables decrease with	**Process** (1 variable)	*Activity variable:* IO Other Bytes/sec (Sa↑C↑)
		Paging File	*State variable:* %Usage

	At. • All variables have the inverse-V relation with E. • The process variable increases with Sa and C.	(1 variable)	
3	At↓E↓K↓Sa↑C↑: • 1 variable in total • The variable decreases with At, E and K. • The variable increases with Sa and C.	**IP** (1 variable)	*Performance variable (QoS):* **Fragments Created/sec**
4	At↓EvKv (SaΛC↑): • 17 variables in total • All variables decrease with At. • All variables have the v-shape relation with E and K • 11 variables have the inverse-V relation with Sa and increase with C. • The above 11 variables and all Process variables (15 variables) have the inverse-v relation with B when there is no data encryption, the v-shape relation with B when there is 50% data encryption, and decrease with B when there is 100% data encryption.	Terminal Service Session (4 variables)	*State variables:* Working Set (SaΛC↑), Page File Bytes (SaΛC↑), Private Bytes (SaΛC↑), Virtual Bytes (SaΛC↑)
		Memory (3 variables)	*State variables:* Committed Bytes (SaΛC↑), %Committed Bytes (SaΛC↑), Pool Paged Bytes
		Objects (5 variables)	*Activity variables:* Processes, Threads (SaΛC↑), Events (SaΛC↑), Mutexes (SaΛC↑), Semaphores (SaΛC↑)
		Process (4 variables)	*State variables:* Virtual Bytes, Working Set, Page File Bytes, Private Bytes
		System (1 variable)	*Activity variables:* Processes (SaΛC↑)
5	At↓: • 1 variable in total. • The variable decreases with At.	**Memory** (1 variable)	*State variables:* System Code Resident Bytes
6	At↑E↓K↓Sa↓C↓: • 5 variables in total • All variables increase with At. • All variables decrease with E, K, Sa and C	**System** (3 variables)	*Activity variables:* File Read Operations/sec, File Write Operations/sec, File Data Operations/sec
		Network (2 variables)	*Activity variables:* Bytes Total/sec, Packets/sec
7	At↑E↑Sa↑C↑: • 2 variables in total	Web Service (1 variable)	*Activity variables:* Current ISAPI Extension Requests

	• All variables increase with At, E, Sa and C.	Memory (1 variable)	*Activity variable:* Cache Faults/sec
8	At↑: • 3 variables in total • All variables increase with At.	**Network** (1 variables)	*Activity variables:* Packets Received/sec
		Server (1 variable)	*Activity variable:* Pool Nonpaged Bytes
		System (1 variable)	*State variable:* %Registry Quota In Use
9	E↓: • 15 variables in total. • All variables decrease with E.	**Physical Disk** (6 variables)	*Activity variables:* Disk Transfers/sec, Disk Writes/sec, Disk Write Bytes/sec, Transition Faults/sec, Write Copies/sec *State variables:* Avg. Disk Write Queue Length
		Web Service (5 variables)	*Activity variables:* Bytes Sent/sec, Files Sent/sec, Files/sec, Bytes Received/sec, Bytes Total/sec
		Process (1 variable)	*Activity variable:* Handle Count
		IP (1 variable)	*Activity variable:* Fragmented Datagrams/sec
		UDP (1 variable)	*Activity variable:* Datagram/sec
		Processor (1 variable)	*Activity variable:* %DPC Time
10	E↑: • 9 variables in total. • All variables increase with E.	**Memory** (3 variables)	*Activity variables:* Pages/sec, Pages Input/sec, Page Reads/sec
		Physical Disk (2 variables)	*Activity variables:* Avg. Disk sec/Transfer, Avg. Disk sec/Write
		Terminal Service Session (1 variable)	*State variable:* %Processor Time
		System (1 variable)	*State variable:* Processor Queue Length
		Physical Disk (1 variable)	*State variables:* Current Disk Queue Length
		Terminal Service Session (1 variable)	*State variable:* Pool Nonpaged Bytes

- Category 1 (At↓E↑Sa↑C↑(K↓), file operations, bytes and page faults of the voice communication service and the data encryption service): The variables reflect privileged file-related IO (both input and output) activities and page faults of voice data communication and data encryption, and increase their values with Sa, C and E. That is, the increasing levels of Sa, C and E produce more file-related activities and page faults of voice data communication and data encryption. Most of the variables decrease their values with K because the use of a larger key length causes an increased computation time of data encryption and thus less IO activities of processes. All variables decrease with At due to the competition of the cyber attack with the voice communication and data encryption services for the CPU time. The attacks take some CPU time from the voice communication and data encryption services, and thus reduce file-related IO activities of these services.
    - *Implications in QoS adaptation, system survivability, load balancing, and attack detection:* the variables in this category indicate the activity level of the voice communication service and the data encryption service. They keep increasing under all the levels of the service parameters in the experiment without reaching any saturation point. These variables can be used to measure the activity level of voice data communication and data encryption.
- Category 2 (At↓E∧(Sa↑C↑), Use of the paging file by the voice communication service and the data encryption service): the variables indicate the use of the paging file by the voice communication service and the data encryption service. The variables decrease with the attack due to the competition for CPU time. The counter IO Other Bytes measuring file-related IO activity, similar to Category 1, increases with Sa, C and E (from level 1 to level 2), but can be saturated by E from level 2 to level 3. The inverse-V (∧) relation with E indicates that the variables in this category first increase their values as E changes from level 1 to level 2 but then decrease their values as E changes from level 2 to level 3.
    - *Implications in QoS adaptation and attack detection:* similar to those for Category 1.
- Category 3 (At↓E↓K↓Sa↑C↑, network throughput of the voice communication service): The IP variable in Category 2, Fragments Created/sec_IP, measures the network data sent and

thus reflects the network throughput of the voice communication service since only the voice communication service sends large amounts of data over the network from the server to clients. The variable increases with Sa and C as increasing levels of Sa and C result in more network data sent over the network. The variable decreases with E and K by increasing encryption percentage and key length data encryption takes more computation time and thus leaves less CPU time available and generates less network throughput for the voice communication service. The variable also decreases with At due to the competition of the cyber attacks with the voice communication service for the CPU time.

  - *Implications in QoS adaptation and attack detection:* Fragments Created/sec_IP can be used as the QoS measure of network throughput for the voice communication service. Both the data encryption service and the attacks degrade the QoS performance of the voice communication service. Hence, when an attack is present and the network throughput of the voice communication service need to be maintained at a certain level, the data encryption service may need to be sacrificed to a certain degree.

- Category 4 (At↓EvKv(SaΛC↑), process and thread counts and memory usage of the voice communication service and the data encryption service): The variables reflect memory usage and counts of processes, threads, events, and so on for the voice communication service and the data encryption service. These variables decrease with At due to the competition of the cyber attacks with the voice communication service and the data encryption service for the CPU time. The variables have the V relation with E and K by first decreasing as E and K change from level 1 to level 2 and then increase as E and K change from level 2 to level 3. That is, the longer computation time of the data encryption caused by the increase in the encryption percentage and the key length from level 1 to level 2 first reduces the number of processes and threads and the memory usage by fewer processes and threads. As the encryption percentage and the key length increases from level 2 to level 3, the processing need of data encryption becomes high enough, leading to the creation of more processes, threads, and so on to handle the larger processing need and thus produce more memory usage by the increasing number of processes and threads. Some variables measuring the process and thread counts and the memory usage also increase with C to reflect the increasing

activity of the voice communication service. These variables also have an inverse V-shape with Sa due to the saturation in the number of processes and threads and the memory usage. That is, the increase of Sa from level 1 to level 2 first increases the process and thread counts and the memory usage. As the number of processes and threads and the memory usage become saturated, the process and thread counts and the memory usage stop increasing and even decrease due to the resource saturation. Hence, Sa has a larger effect on the number of processes and threads and the memory usage than C.

  - *Implications in QoS adaptation and attack detection:* Committed Bytes_Memory for both the voice communication service and the data encryption service is a key state variable in this Category. The memory usage by both services decreases when the attacks occur due to competition for CPU time, has the V relation with E and K, increases with C, and has the inverse-V relation with Sa. This implies that using level 2 of the data encryption percentage and the key length for the data encryption service will be optimal in minimizing the memory usage by the data encryption service. There is less room for increasing the sampling rate of the voice communication service due to the memory constraint. Increasing the number of clients to even the highest level in the experiments does not present a major problem in memory usage.

- Category 5 (At↓, system code resident bytes in memory for the voice communication service and the data encryption service): The variables reflect system code resident bytes in memory used by the voice communication service and the encryption service, which decrease with the attack due to competition of the attack for CPU time. However, such system code of the voice communication service and the data encryption service does not change with more activities of these two services.
  - *Implications in QoS adaptation and attack detection:* the variable in Category 5 does not change its values with parameters of the voice communication service and the data encryption service but with the attacks only. Hence, this variable can be used to detect the attacks through decreased values.
- Category 6 (At↑E↓K↓Sa↓C↓, file operations with no change in bytes and network bytes and packets related to the cyber attacks): the variables reflect file operations and network bytes

and packets affected by the cyber attacks. The attacks affect file operations but not bytes of file operations. These variables decrease with Sa, C, E and K due to competition for CPU time among the cyber attacks, voice data communication and data encryption.

  - *Implications in QoS adaptation and attack detection:* The cyber attacks, voice data communication and data encryption compete for CPU time. When an attack occurs and is detected, the activity level of voice data communication and data encryption can be increased to take away more CPU time from the attack and sustain the performance level of voice data communication and data encryption.

- Category 7 (At↑E↑Sa↑C↑, cache faults and Current ISAPI Extension Requests of web service increased by the attacks, voice data communication and data encryption): cache faults and Current ISAPI Extension Requests of web service are increased by all three activities and possibly any other activities.
  - *Implications in QoS adaptation and attack detection:* Since these activity variables are increased by all types of activities, they can be used to indicate the overall system load by everything going on the system.
- Category 8 (At↑, activity of the attack): the variables measure received network packets, pool nonpaged bytes, and %registry quota in use, which are affected by the attacks only.
  - *Implications in QoS adaptation and attack detection:* the variables in Category 8 do not change their values with parameters of the voice communication service or the data encryption service but with the attacks only. Hence, the variables in Category 8 can be used to detect the attacks through increased values.
- Category 9 (E↓, variables affected by the data encryption service only due to competition for CPU time): the variables reflect disk writes of the physical disk caused by the attack and network activities of the voice communication service. These variables are not significantly affected by the attacks and are not consistently affected by the parameters of the voice communication service. These variables decrease with E due to more computation time taken by data encryption, that is, the competition of the data encryption service with the voice communication service and the attack for CPU time.
  - *Implications in QoS adaptation and attack detection:* these variables reflect

competition of the attacks, voice communication service and the data encryption service for CPU time, which implies the tradeoff among the three activities and implications as indicated for Category 6.

- Category 10 (E↑, activities of the data encryption service in memory and physical disk): the variables reflect page read and input in memory and associated disk activities of the physical memory as well as CPU utilization by the data encryption service. They increase with more activities of the data encryption service. The attack has little effect on these variables, which are not consistently affected by the voice communication service.
  - *Implications in QoS adaptation and attack detection:* these variables indicate the activity level of the data encryption service, which does not reach the saturation under the experimental conditions.

Dynamics effects of the voice communication service. The voice communication service involves significantly file operations and bytes, page faults and cache faults, use of the paging file, processes/threads/events created, memory usage, and network data sent as shown in Figure 6. These activity, state, and performance variables increase with Sa and C. The memory usage and creation of processes/threads/events can be saturated by a higher level of Sa. Caution should be taken when increasing the sampling rate of voice data communication due to the memory constraint. Increasing clients does not present problems to resources.

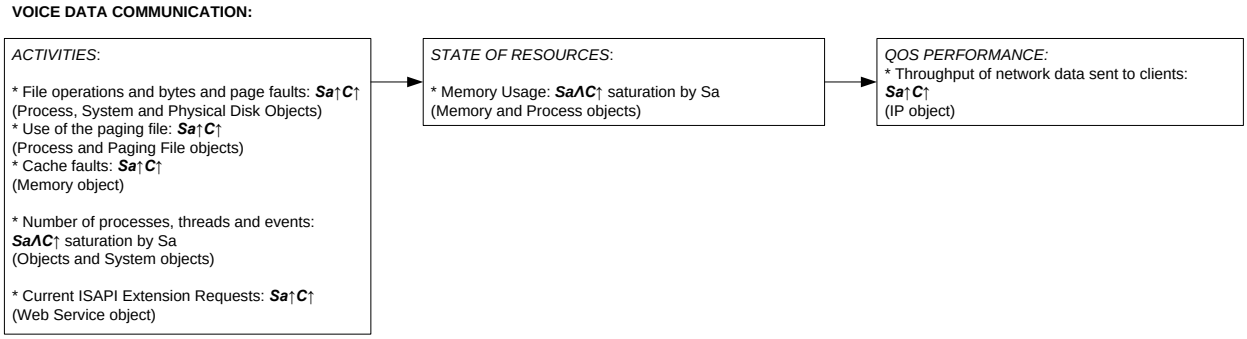


Figure 6: Dynamics Effects of the Voice Communication Service

Dynamic effects of the data encryption service. As shown in Figure 7, the data encryption

service involves significantly file operations and bytes, page reads and inputs, page faults, and disk writes in physical disk, processes/threads/events created, memory usage, and CPU utilization. These variables keep increasing with more percentage of data encryption service. The longer computation time of the data encryption caused by the increase in the encryption percentage and the key length from level 1 to level 2 first reduces the memory usage and the process and thread counts by fewer processes and threads. As the encryption percentage and the key length increases from level 2 to level 3, the processing need of data encryption becomes so high to drive the creation of more processes, threads, and so on to handle the larger processing need and thus produce more memory usage by the increasing number of processes and threads. The process and thread counts and the memory usage do not reach the saturation point at the highest level of E and K.

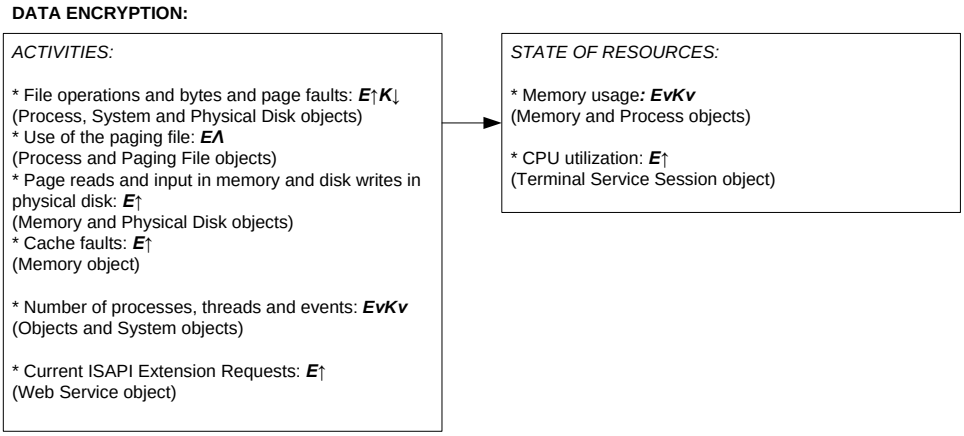


Figure 7: Dynamics Effects of the Data Encryption Service.

Dynamics effects of attacks. Figure 8 shows dynamic effects of attacks which are common and consistent in both the voice communication service experiments and the motion detection experiments. The attacks involve paging, cache faults, thread count, handle count, exception dispatches, web service connections, and memory usage. **A**ttacks increase the threads count and exception dispatches, and increase pages to and from the hard disk, the memory usage and cache faults because attack activities trigger more intrusion detection activities. The attacks decrease the use of the working set, sections, cache bytes, pool paged and nonpaged byes, handle count,

and web service activities, due to the competition for CPU time between the attacks and motion detection service.

Tradeoff among voice data communication, data encryption and attacks. The three activities, voice data communication, data encryption, and attacks, compete for CPU time. An increase in one activity affects the two other activities. Data encryption affects CPU utilization more than voice data communication and attacks. When an attack occurs and is detected, the activity level of voice data communication and data encryption can be increased, e.g., increasing S and C of voice data communication, to take away more CPU time from the attack and sustain the performance level of voice data communication and data encryption. When an attack is present and the network throughput of the voice communication service need to be maintained at a certain level, the data encryption service may need to be sacrificed to a certain degree. Cache faults and Current ISAPI Extension Requests of web service are increased by all three activities and possibly any other activity, and can be used to indicate the overall system load by everything going on the system.

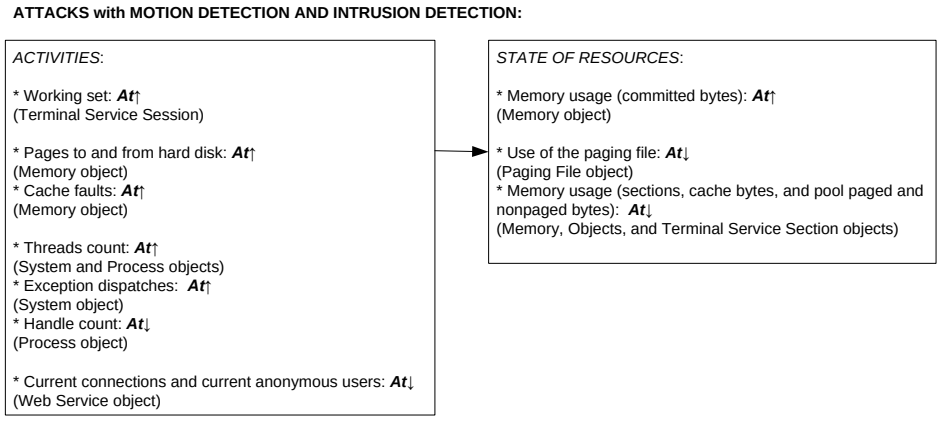


Figure 8: Dynamics Effects of Attacks.

Windows performance objects. Among the Windows performance objects, the following objects contain variables that reflect activities of the voice communication service, the data encryption service, and the attacks: Process, IP, Network, Memory, Physical Disk, System and Objects. Server activities are shown through variables of the Process object. Other objects such

as Terminal Service Session and Processor contain similar information. Terminal Service Session contains information similar to that in Process.

2.3.2 Dynamic Effects and Tradeoffs of the Motion Detection Service, the Intrusion Detection Security and the Attacks

Table 9 shows the types of relations and tradeoff effects of the experimental parameters with system dynamics variables for the motion detection service, the intrusion detection service, and the cyber attacks. (R: Resolution of video; T: Threads; D: Detection of intrusions by SNORT)

Table 9: System dynamics for the motion detection service, the intrusion detection service, and the cyber attacks: Categories

Category	Characteristics of Relations and Tradeoff Effects	Object	Variables
1	At↑ID↑T↑: • 4 variables in total. • All variables increase with At, ID and T.	**System** (2 variables)	*Activity variables:* Exception Dispatches/sec, Threads
		Memory (2 variables)	*State variables:* Committed Bytes, %Committed Bytes
2	At↑ID↓T↑Rv: • 1 variables in total. • The variable increases with At and T, decreases with ID, and has a v-shape relation with VS.	Terminal Service Session (1 variable)	*State variable:* Working Set
3	At↓ID↑T↓: • 3 variables in total • All variables decrease with At and T, and increase with ID.	**Memory** (3 variables)	*State variables:* Available Bytes, Available KBytes, Available Mbytes
4	At↑ID↑T↓: • 2 variables in total. • All variables increase with At and ID, and decrease with T.	**Memory** (2 variables)	*Activity variables:* Pages/sec, Cache Faults/sec

5	At↓D↑T↑: • 3 variables in total. • All variables decrease with At, and increase with ID and T.	Web Service (2 variables)	*Activity variables:* Current Connections, Current Anonymous Users
		Paging File (1 variable)	*State variable:* % Usage
6	At↓ID↓T↑: • 5 variables in total • All variables decrease with At and ID, and increase with T.	**Process** (1 variable)	*Activity variable:* Handle Count
		Memory (3 variables)	*State variables:* System Cache Resident Bytes, Pool Paged Bytes, Pool Paged Resident Bytes
		Objects (1 variable)	*Activity variable:* Sections
		Terminal Service Session (1 variable)	*State variable:* Pool Nonpaged Bytes
7	R↑ • 1 variable in total • Motion level increases from level 1 to level 2 of Video Resolution	Performance (1 variable)	*Performance variable:* Motion level

- Category 1 (At↑ID↑T↑, the memory usage, the threads count and exception dispatches used by motion detection, intrusion detection, and the attacks): all three activities increase memory usage, threads count and exception dispatches of such code.
  - *Implications in QoS adaptation and attack detection:* Since these variables are increased by all types of activities, they can be used to indicate the overall activity level and system load by everything going on the system.
- Category 2 (At↑ID↓T↑Rv: working set used by motion detection and the attacks): motion detection and attacks use the working set, whereas intrusion detection uses the hard disk to retrieve intrusion signatures. The competition of intrusion detection with motion detection and the attacks for CPU time reduces the working set when intrusion detection runs. When the video resolution increases from level 1 to level 2, motion detection takes more computation and thus reduces working set. The increase of video resolution from level 2 to level 3 produces more intermediate results, and thus causes the working set to increase.
- Category 3 (At↓ID↑T↓, available memory): Category 3 is opposite to the memory usage by

in Category 1. The available memory increases with ID since ID takes data from the hard disk instead of the memory.

- Category 4 (At↑ID↑T↓: cache faults and pages read from and written to hard disk of intrusion detection): the hard disk usage and cache faults by intrusion detection, which increase with intrusion detection. They increase with the attacks since intrusion detection is more active to analyze attack activities. These variables decrease with T due to competition for CPU time between intrusion detection and motion detection.
  - *Implications in QoS adaptation and attack detection:* these variables indicate the activity level of intrusion detection.
- Category 5 (At↓ID↑T↑: use of the paging file and web service by motion detection): motion detection uses the web service and the paging file. These variables increase with intrusion detection because intrusion detection also analyzes the web service activities of motion detection. These variables decrease with the attacks due to competition for CPU time between motion detection and the attacks.
  - *Implications in QoS adaptation and attack detection:* The variables can be used to measure the activity level of motion detection.
- Category 6 (At↓ID↓T↑, handle count, sections, system cache bytes, pool paged and nonpaged bytes in memory of motion detection): these variables measure activities and memory usage by motion detection. They increase with T, but decrease with At and ID due to competition for CPU time.
  - *Implications in QoS adaptation and attack detection:* these variables can be used to measure the activity level of motion detection.
- Category 7 (R↑, motion detection level): the higher video resolution, the higher motion detection level. Motion level is not directly affected by the activities of the intrusion detection nor the activities of the attack.

Dynamics effects of the motion detection service. Motion detection service involves significantly memory usage, paging, cache faults, threads, handle counts, exception dispatches, and web service connections as shown in Figure 9. As motion detection threads increase, threads

and handle counts increase along with the working set used by these threads. Since these threads use the same video file and share the same cache and memory content, the memory usage increases and pages read from and written to the hard disk and cache faults decrease as there are more motion detection clients. Motion detection uses the web service. Hence, current connections and anonymous users of web service increases with motion detection threads. The motion level measures how much motion is detected. Hence, the motion level increases as the video resolution increases from level 1 to level 2. However, increasing the video resolution further to level 3 does not improve the motion level significantly.

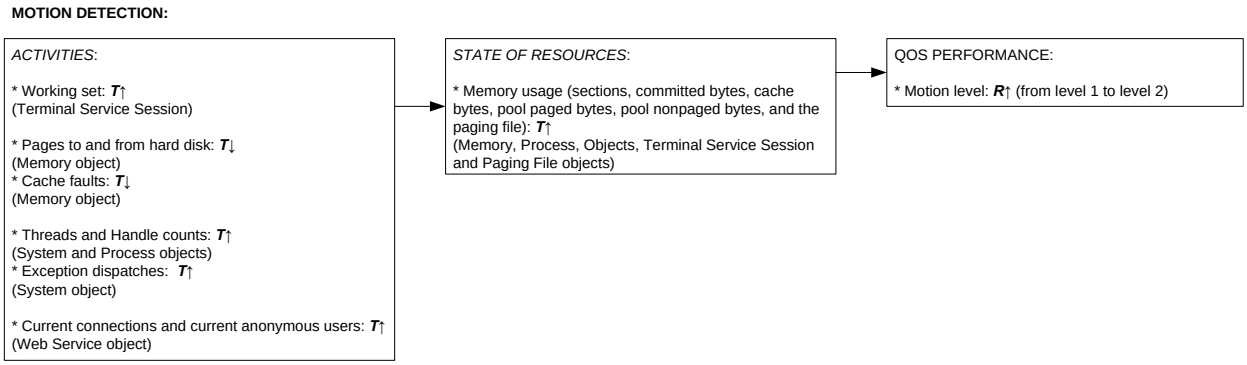


Figure 9: Dynamics Effects of the Motion Detection Service.

Dynamic effects of the intrusion detection service. Intrusion detection service (ID) involves significantly paging, cache faults, thread count, handle count, exception dispatches, web service connections, and memory usage as shown in Figure 10. ID access intrusion signatures from the hard disk and write logs to the hard drive continuously. Hence, pages read from and written to the hard disk, the memory usage and cache faults increase when intrusion detection runs. ID increases the threads count and exception dispatches. ID activity increases with the activity level of both the attacks and motion detection service since ID analyzes the activities of both attacks and motion detection. The working set and the handle count used by motion detection decreases with the running of ID due to competition for CPU time.

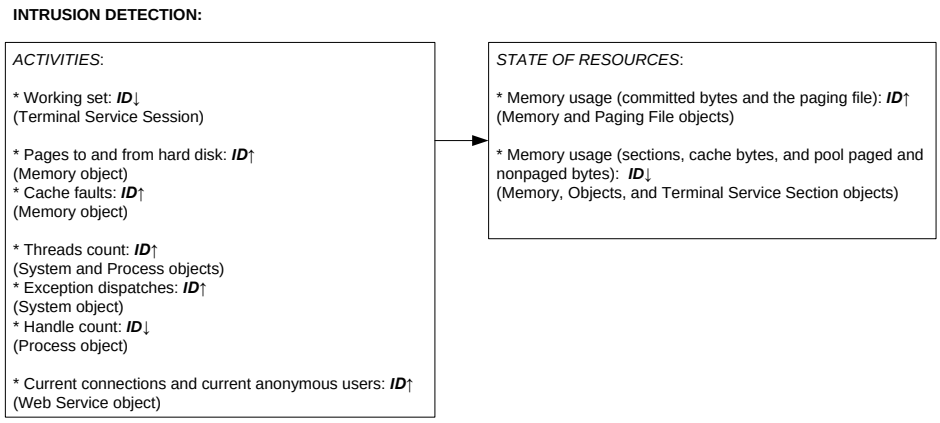


Figure 10: Dynamics Effects of the Intrusion Detection Service.

Dynamics effects of the attacks. See the description in Section 2.3.1.

Tradeoff of motion detection, intrusion detection and attacks. Intrusion detection activities increase with motion detection and attack activities because intrusion detection analyzes motion detection and attack activities. There is competition for CPU time between motion detection and the attacks.

Windows performance objects. Among the Windows performance objects, the following objects contain variables that reflect activities of the voice communication service, the data encryption service, and the attacks: Process, Memory, System and Objects. Web Service object is used because motion detection uses the web service.

2.3.3 Common and Unique Effects of Various Attacks

The following variables show a consistent decrease effect with attacks in all the experiments.

1. Sections_Objects
2. %Usage_Paging File
3. System Code Resident Bytes_Memory

4. Virtual Bytes_Process.

These variables reflect the activity level of voice data communication and motion detection. They decrease with the attacks due to the competition of the attacks with voice data communication and motion detection for CPU time.

The following variables show a consistent increase effect with attacks in all the experiments.

1. Packets Received Non-Unicast/sec_Network
2. Packets Received/sec_Network

These variables reflect the activity level of the attacks, four of which involve received network data. Received network data increases when the attacks occur.

The following variables show a characteristic increase which is unique to the NMAP attack in all the experiments.

1. Get Requests/sec_Web Service
2. Not Found Errors/sec_Web Service.

These variables can be used to detect the NMAP attack.

The following variables show a characteristic increase which is unique to the Remote Dictionary attack in all the experiments.

1. Bytes Received/sec_Redirector (Increase)
2. Bytes Total/sec_Redirector (Increase)
3. Bytes Transmitted/sec_Redirector (Increase)
4. Packets Transmitted/sec_Redirector (Increase)
5. Read Packets Small/sec_Redirector (Increase)
6. Read Packets/sec_Redirector (Increase)
7. Write Packets Small/sec_Redirector (Increase)
8. Write Packets/sec_Redirector (Increase)

These variables can be used to detect the Remote Dictionary attack.

2. 4 Conclusions

We present our methodology for uncovering dynamic effects and tradeoffs of service, security and attack activities required for QoS and QoIA adaptation, system survivability and resource load balancing. The methodology is illustrated through the experiments involving two specific scenarios: an encrypted voice communication service and a network-secured motion detection service. System dynamics data is collected for both scenarios under various service and security conditions. Cyber attacks such as ARP Poison, ping flood, NMAP, fork bomb, and remote dictionary are launched over the server while running the service scenarios. By performing statistical analysis on the data collected, dynamic effects and tradeoffs among services, security mechanism and cyber attacks are uncovered. The discovered information enhances our understanding of the system dynamics and the interactions between service, security mechanism and attacks.

For the encrypted voice communication scenario, the system dynamics created by the voice communication service, the data encryption service, and the cyber attacks were grouped into 10 categories. The characteristics of the relations in each category, tradeoff effects and their implications for QoS adaptation, system survivability and load balancing, if any, are provided.
For example from the analysis on the counter *Fragments Created/sec* of the IP object in Category 3, used as the QoS measure of network throughput for the voice communication service, when an attack is present and there is no resources available in the system to maintain the network throughput of the voice communication service at a certain level the data encryption service may be sacrificed to a certain degree to keep providing the desired network throughput.

For the network-secured motion detection service, the system dynamics created by the motion detection service, the intrusion detection service, and the cyber attacks were grouped into 7 categories. The characteristics of the relations in each category, tradeoff effects and their implications for QoS adaptation, system survivability and load balancing, if any, are provided. For example, intrusion detection activities increase with motion detection and attack activities as intrusion detection analyzes motion detection and attack activities, which implies a fierce

competition for resources by the MDS, IDS and attacks, especially for the CPU required mostly by the MDS and the attacks.

Additionally, system dynamics variables showing unique attacks effects regardless of the service and security mechanism running on the system were discovered (Section 2.3.3). These variables can be used by the system to detect attacks. Some of these variables present unique characteristics to NMAP and Remote Dictionary attacks.

In addition to the specific findings of ASQ dynamics and models for specific services, security mechanisms and attacks and implications of these findings for QoS and QoIA adaptations, a general approach to QoS and QoIA adaptations is found to raise or lower the activity level of services, security mechanisms and attacks by utilizing their competition for the limited resources such as CPU time, memory and network bandwidth. Because all activities on a computer and network system share and compete for the limited system resources and the round-robin method of resource sharing is usually used on computers and networks, an increased level of one activity causes a decreased level of other activities sharing the resources. On one hand, this can mean a negative impact of attack activities on regular service and security activities as the increased level of attack activities reduce the level of service and security activities due to their sharing and competition for system resources. On the other hand, this sharing and competition for limited system resources can be used in a positive way for service and security activities. Instead of letting attack activities take more system resources, services and security mechanisms can increase their activity level when an attack occurs. The increased demand level of services and security mechanisms will allow them to increase their activity levels and obtain more system resources, thus suppressing the level of attack activities and sustaining QoS and QoIA levels during an attack. This general finding about QoS and QoIA adaptations is contradictory to what we commonly believe that attacks simply take away more system resources. System resources

taken away by attacks can be taken back by regular services and security mechanisms by increasing the demand and thus activity level of services and security mechanisms for QoS and QoIA adaptations and survivability of services and security mechanisms running on a computer and network system under attacks.

References

Abdel-qader, F. M. (2007). *Examples to create your conferencing system in .NET, C# VOIP & video conferencing systems using H.323 and TAPI 3*. Retrieved 03/16, 2009, from http://www.codeproject.com/KB/IP/Video_Voice_Conferencing.aspx?fid=373359&df=90&mpp=25&noise=3&sort=Position&view=Quick&select=2645686

Atighetchi, M., Pal, P., Webber, F., Schantz, R., Jones, C., and Loyal, J. (2004). Adaptive cyberdefense for survival and intrusion tolerance, *IEEE Internet Computing*, vol. 8, no. 6, pp. 25-33.

Daemen, J., and V. Rijmen (2001). *The Design of Rijndael: AES- The Advanced Encryption Standard*. Springer-Verlag New York Inc.: NJ, USA.

Insecure.Org. *Nmap.Org*. Retrieved 10/10, 2010, from http://nmap.org/

Kirillov, Andrew (2007). *Motion Detection algorithms*. Retrieved 10/09, 2010, from http://www.codeproject.com/KB/audio-video/Motion_Detection.aspx

Lipson, H. F., and Fisher, D. A. (1999). Survivability – a new technical and business perspective, *Proceedings of the 1999 Workshop on New Security Paradigms*, pp. 33-39.

Microsoft. (2003). *Window server 2003 performance counters reference*. Retrieved 03/01, 2009, from http://technet2.microsoft.com.ezproxy1.lib.asu.edu/windowsserver/en/library/3fb01419-b1ab-4f52-a9f8-09d5ebeb9ef21033.mspx

Mitre Corporation. *CAPEC-1000: Mechanisms of Attack*. Retrieved 04/02, 2010, from http://capec.mitre.org/data/graphs/1000.html

Montoro, Massimiliano. *Cain & Abel software v4.9.30*. Retrieved 10/10, 2010, from http://www.oxid.it/index.html

Sourcefire, Inc. *Snort*. Retrieved 10/09, 2010, from http://www.snort.org/

Tools4ever. *Free Ping v2.0*. Retrieved 10/10, 2010, from http://www.tools4ever.com/

Xiao, K., Chen, N., Ren, S., Shen, L., Sun, X., Kwiat, K., and Macalik, M. (2007). A workflow-

based non-intrusive approach for enhancing the survivability of critical infrastructures in cyber environment, *Proceedings of the 3rd International Workshop on Software Engineering for Secure Systems*, pp. 20-26.

Ye, N. (2002). QoS-centric stateful resource management in information systems, *Information Systems Frontiers*, vol. 4, no. 2, pp. 149-160.

Ye, N., Newman, C., and Farley, T. (2005). A system-fault-risk framework for cyber attack classification, *Information, Knowledge, Systems Management*, vol. 5, no. 2, pp. 135-151.

Ye, N. (2008). *Secure computer and network systems: Modeling, analysis and desig*n Wiley Publishing: London, UK.

Yi, X. and Zhang, Y. (2005). *Survivability of information system*," *Proceedings of the 5th International Conference on Information, Communications and Signal Processing*, pp. 1551-1555.

Zhang, L.J., Wang, W., Guo, L., Yang, W., and Yang, Y. T. (2007). A survivability quantitative analysis model for network system based on attack graph, *Proceedings of the 6th International Conference on Machine Learning and Cybernetics*, pp. 19-22.

Zuo, Y. and Panda, B. (2009). Unifying strategies and tactics: a survivability framework for countering cyber attacks, *Proceedings of the IEEE International Conference on Intelligence and Security Informatics*, pp. 119-124.

List of Abbreviations and Acronyms

At	Cyber **At**tacks
ARP	**A**ddress **R**esolution **P**rotocol
ANOVA	**An**alysis **o**f **Va**riance
ASQ	**A**ctivity-**S**tate-**Q**uality
B	**B**uffer size
C	Number of **C**lients
CPU	**C**entral **P**rocessing **U**nit
DOS	**D**enial **o**f **S**ervice
E	**E**ncryption percentage
HSD	**H**onest **S**ignificant **D**ifference
ID	**I**ntrusion **D**etection service
IO	**I**nput/**O**utput
IP	**I**nternet **P**rotocol
K	**K**ey length
MDS	**M**otion **D**etection **S**ervice
QoS	**Q**uality **o**f **S**ervice
R	**R**esolution
Sa	**Sa**mpling rate
SS	**S**ecurity **S**ervice
T	Number of **T**hreads/Clients
TCP	**T**ransmission **C**ontrol **P**rotocol
TSS	**T**erminal **S**ervice **S**ession
UDP	**U**ser **D**atagram **P**rotocol
VCS	**V**oice **C**ommunication **S**ervice